Large Scale file storage with MogileFS

Stuart Teasdale
Lead System Administrator
we7 Ltd

About We7

- A web based streaming music service
- 6.5 million tracks
- 192kbps and 320kbps mp3s
- Sending over a gigabit a second of streams at peak

Our requirements

- Store all those mp3s
- Be able to grab any stream quickly
- Losing files not acceptable
- Files being offline not acceptible

First Solution

- Big NFS Server
- RAID5
- Another one in a different datacentre
- rsync between the two

Problems with NFS/RAID

- Does anyone trust RAID5 any more?
- Increasing Capacity is tough
- NFS fragility
- All eggs in one software and hardware basket

Options

- Big old Sun Fire machine with 48 1TB disks
  - Extension of the previous solution
  - Shift the scaling limit
  - Disk failure handling
- SANs and NASs
  - Price and power consumption
- Spread it across lots of disks and machines
  - Adds complexity
  - Allows us to get more from our hardware

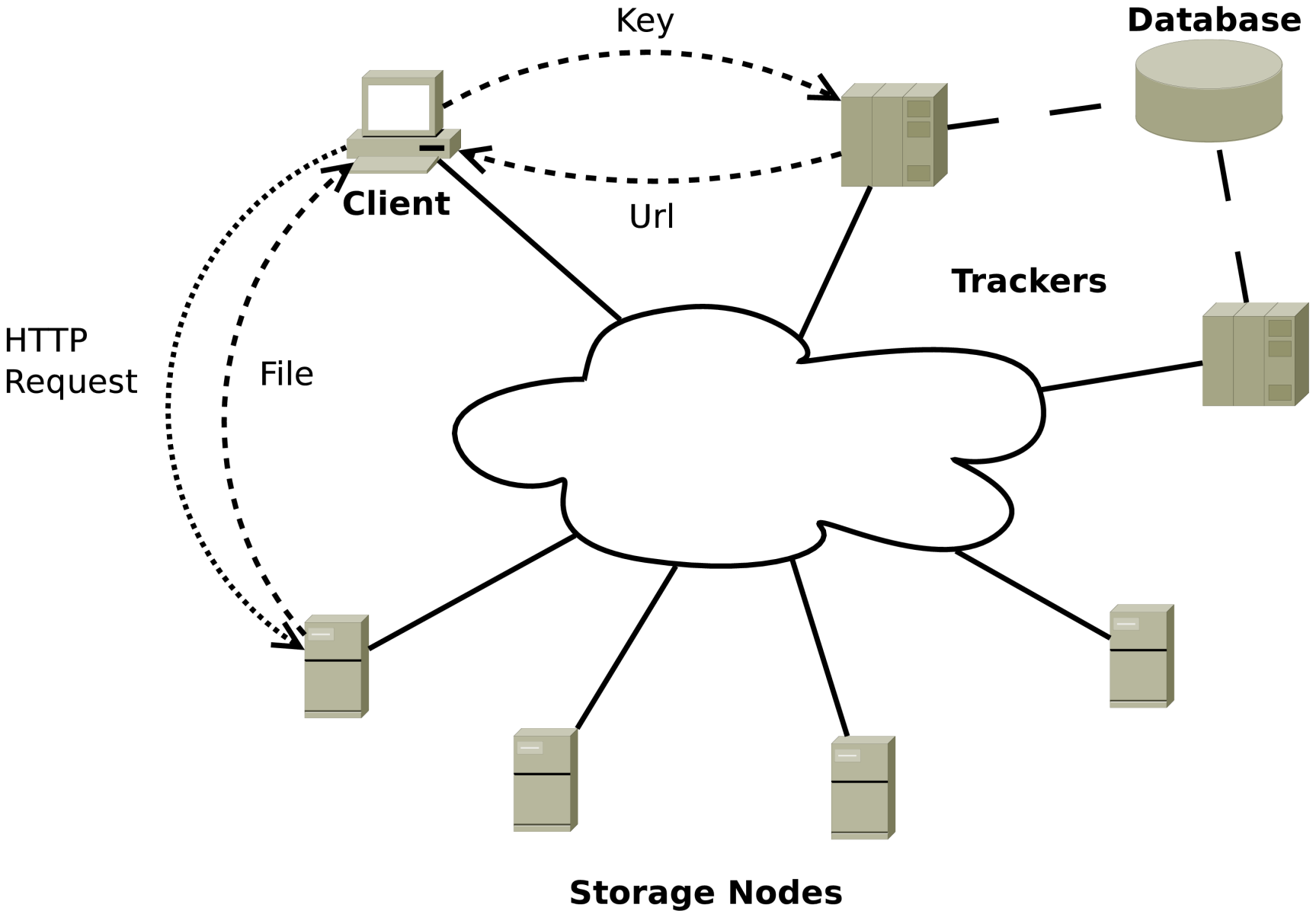
Distributed Filesystems

- Two main types:
  - Full POSIX filesystems
    - ocfs2, gfs
    - Need full clustering, lock managers, etc
  - Application filesystems
    - Throw out all that POSIX hassle
    - Make your application do the hard work
    - Examples: HadoopFS, the other GFS, MogileFS

MogileFS

- Application level
- No single points of failure
- Automatic file replication
- “Better than RAID”
- Flat namespace
- Shared-nothing
- No RAID required
- Local filesystem agnostic

Logical Layout



Storage Nodes

- WebDAV server
  - Built in perlbal, or use lighttpd, nginx, etc
- Storage Node statistics
- Hardware needs:
  - Lots of disks
  - Decent net cards
  - Hotswap disk controller useful
  - Leaves plenty of CPU and memory for other tasks

Trackers

- Manages all client communications
- Parent passes requests to 'query workers'
- Workers do:
  - Replication
  - Deletion
  - Query
  - Reaper
  - Monitor
- Parent can load balancer to multiple workers
- Hardware needs are modest

Database

- Used as metadata store
- Mysql most mature
- Postgres and sqllite also available
- Do HA however you prefer
- Database size is proportionate to number of files stored, but not huge
- Hardware needs in line with typical database app

Domains, Classes and Files

- Domains
  - Top level division for the store
  - File keys unique within a Domain
- Classes
  - Defines groups of files that share replication policies
  - Each file in a domain must belong to one class
- Files
  - The object we actually store

Replication Policies

- Simple: How many copies (mindevcount)
- Complex: Where to put them?
  - Hooks to add custom policies

Zones and Networks

- Example of a more complex replication policy
- Network module makes MogileFS network aware
- Zones module defines different networks as different zones
  - zone_hex = 10.0.2.0/24,zone_sov = 10.0.1.0/24
- Replication policy defines where files are stored
  - HostsPerNetwork(sov=2,hex=2)

Coping with Failure

- Device States
  - Alive, Read-Only, Drain, Down, Dead
- Host States
  - Alive, Down, Dead
- Dead devices and the Reaper

Drain and Rebalance

- Pre 2.40
  - Drain removes files from device
  - Basic rebalancing
- Post 2.40
  - Complex rebalance policies
  - Drain no longer removes files

Checks and Monitoring

- Internal 'fsck' walks all the files and check:
  - File exists where we expect it to
  - Replication policy is fulfilled
- We7 does extra integrity checks on a per server basis
- Monitoring plugins exist for munin and nagios

Using MogileFS – Command Line

- Mogadm
  - Manipulate Domains and Classes
  - Add modify and remove hosts and devices
  - Control cluster settings
  - Control the fsck worker
- Mogstats
  - Show device usage and status
  - Show queue and worker states
  - List files and their replication counts

Using MogileFS – Command Line

- Mogtool
  - Add, remove query and manipulate files
  - Deprecated in favour of more unixlike set of tools
    - mogdelete Delete keys from a MogileFS installation
    - mogfetch Fetch data from a MogileFS installation
    - mogfiledebug Dump gobs of information about a FID
    - mogfileinfo Fetch key metadata from a MogileFS installation
    - moglistfids Iterate fid/key data from a MogileFS installation
    - moglistkeys Lists keys out of a MogileFS domain
    - mogupload Upload data to a MogileFS installation

Language Bindings

- Implemented in Perl, so good Perl client library support
- Java bindings – used by we7
- Client libraries available for Ruby, PHP and Python

Scaling Up

- Adding storage nodes
- Add more trackers
- Read only queries and database slaves
- Application side caching, e.g. in memcached

More Information

- http://www.danga.com/mogilefs/
- http://code.google.com/p/mogilefs/wiki/
- http://groups.google.com/group/mogile

Questions?