Deploying Grid Services Using Hadoop

Allen Wittenauer

April 11, 2008

Agenda

- Grid Computing At Yahoo!
- Quick Overview of Hadoop
  - “Hadoop for Systems Administrators”
- Scaling Hadoop Deployments
- Yahoo!’s Next Generation Grid Infrastructure
- Questions (and maybe even Answers!)

Grid Computing at Yahoo!

- Drivers
  - 500M unique users per month
  - Billions of interesting events per day
  - *"Data analysis is the inner-loop at Yahoo!"*
- Yahoo! Grid Vision and Focus
  - On-demand, shared access to vast pool of resources
  - Support for massively parallel execution (1000s of processors)
  - Data Intensive Super Computing (DISC)
  - Centrally provisioned and managed
  - Service-oriented, elastic
- What We're Not
  - Not "Grid" in the sense of scientific community (Globus, etc)
  - Not focused on public or 3rd-party utility (Amazon EC2/S3, etc)

Yahoo! Grid Services

- Operate multiple grids within Yahoo!
- 10,000s nodes, 100,000s cores, TBs RAM, PBs disk
- Support large internal user community
  - Account management, training, etc
- Manage data needs
  - Ingest TBs per day
- Deploy and manage software (Hadoop, Pig, etc)

“Grid” Computing

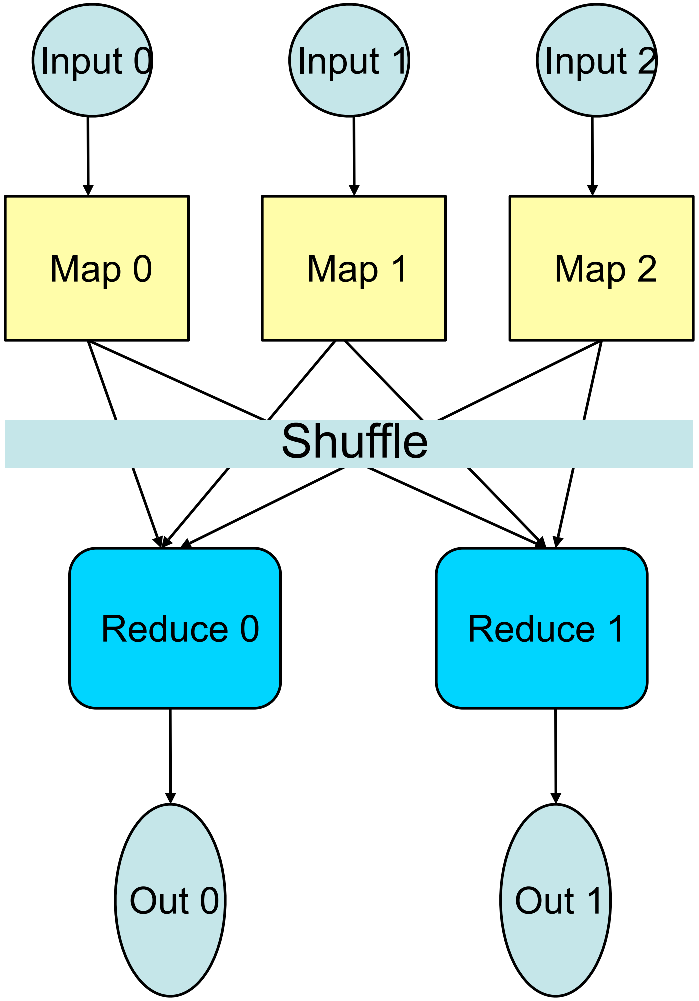
- What we really do is utility computing
  - Nobody knows what “utility computing” is, but everyone has heard of “grid computing”
    - Grid computing implies sharing across resources owned by multiple, independent organizations
    - Utility computing implies sharing one owner’s resources by multiple, independent customers
- Ultimate goal is to provide shared compute and storage resources
  - Instead of going to a hardware committee to provision balkanized resources, a project allocates a part of its budget for use on Yahoo!’s shared grids
  - Pay as you go
    - Only buy 100 computers for 15 minutes of compute time vs. 100 computers 24x7

What is a Yahoo! Grid Service?

- Thousands of machines using basic network hardware
  - It’s hard to program for many machines
- Clustering and sharing software
  - Hadoop, HOD, Torque, Maui, and other bits...
- Petabytes of data
  - It’s an engineering challenge to load so much data from many sources
- Attached development environment
  - A clean, well lit place to interact with a grid
- User support
  - Learning facilitation
- Usage tracking and billing
  - Someone has to pay the bills...

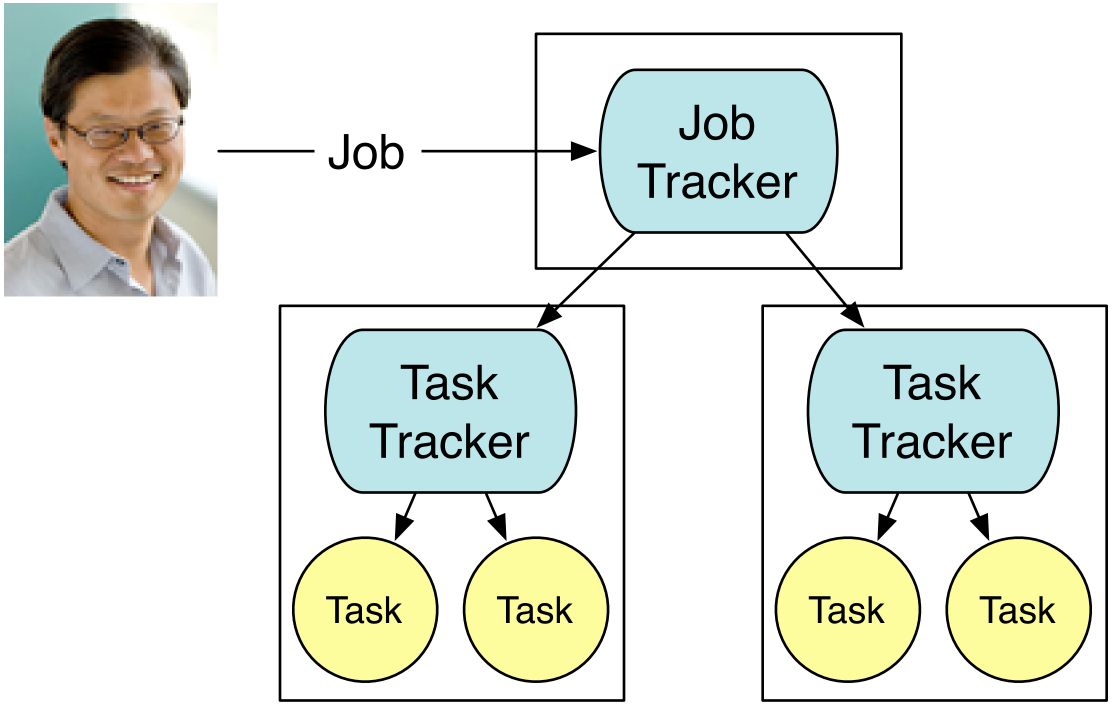
Quick MapReduce Overview

- Application writer specifies
  - two functions: Map and Reduce
  - set of input files
- Workflow
  - input phase generates a number of FileSplits from input files (one per Map Task)
  - Map phase executes user function to transform key/value inputs into new key/value outputs
  - Framework sorts and shuffles
  - Reduce phase combines all k/v's with the same key into new k/v's
  - Output phase writes the resulting pairs to files
- cat * | grep | sort | uniq -c | cat > out

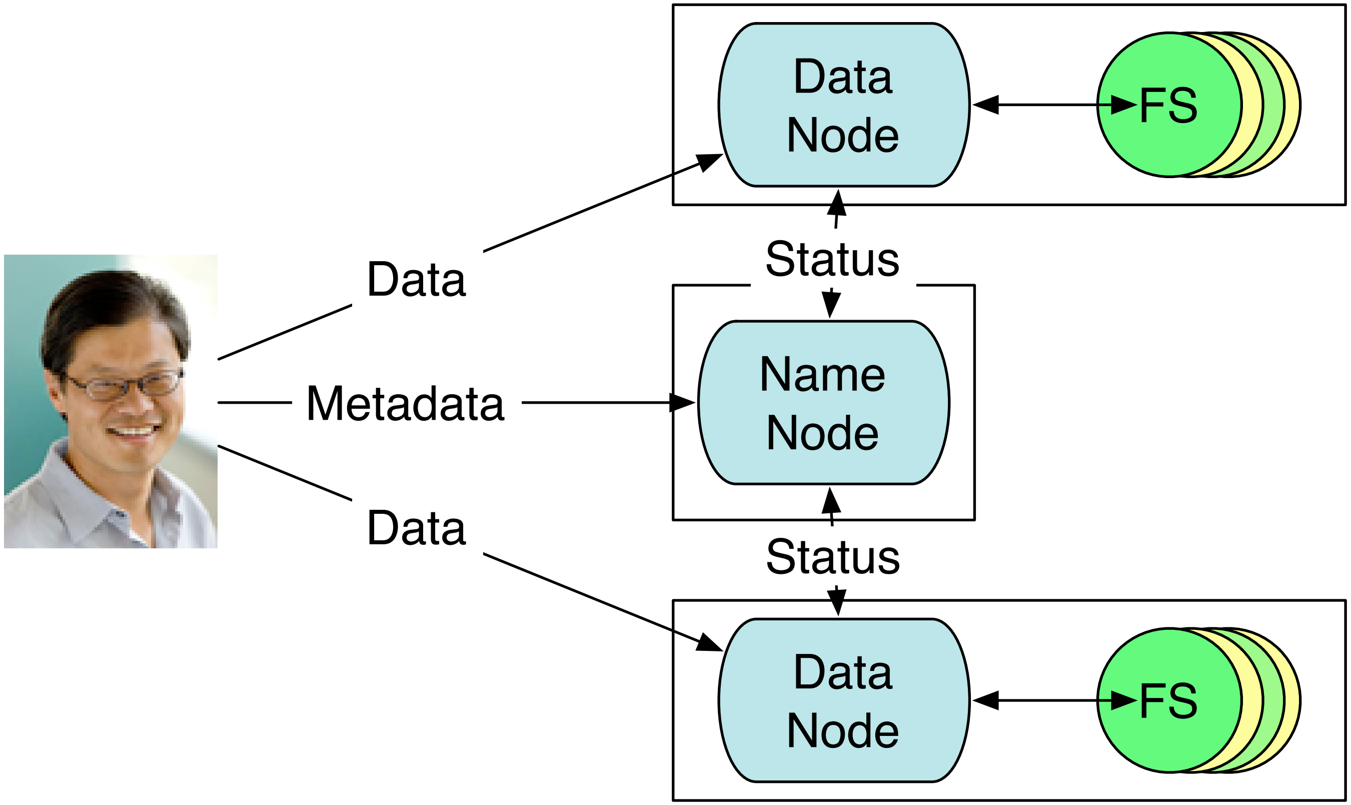


Hadoop MapReduce: Process Level

- Job
  - Map Function + Reduce Function + List of inputs



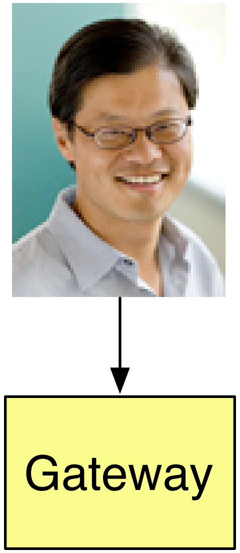
Hadoop Distributed File System



Let op afstap!

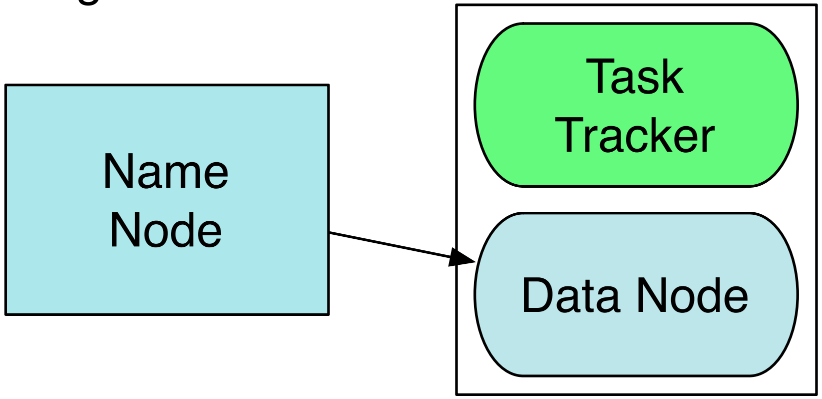
Gateways: Multi-user Land

- Provided for two main purposes
  - Meaningful development interaction with a compute cluster
    - High bandwidth, low latency, and few network barriers enable a tight development loop when creating MapReduce jobs
  - Permission and privilege separation
    - Limit exposure to sensitive data
      - Hadoop 0.15 and lower lack users, permissions, etc.
      - Hadoop 0.16 has users and “weak” permissions
- Characteristics
  - “Replacement” Lab machine
  - World-writable local disk space
    - Any single-threaded processing
    - Code debugging



Compute Cluster and Name Nodes

- Compute Cluster Node
  - Users cannot login!
  - All nodes run both MapReduce and HDFS frameworks
  - usually 500 to 2000 machines
  - Each cluster kept relatively homogenous
  - Hardware configuration
    - 2xSockets (2 or 4 core)
    - 4x500-750G
    - 6G-8G RAM
- Name Node
  - 16G RAM
    - 14G Java heap = 18-20 million files

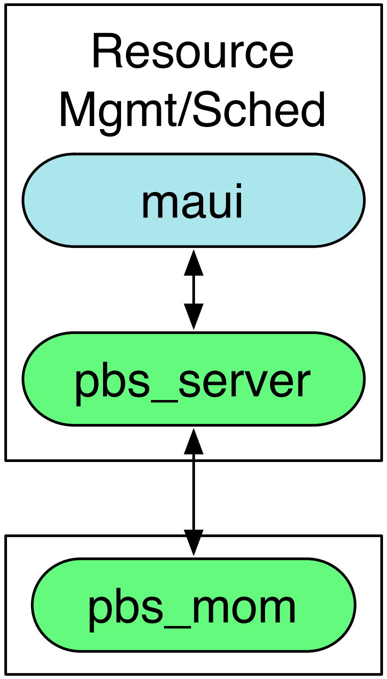


Queuing and Scheduling

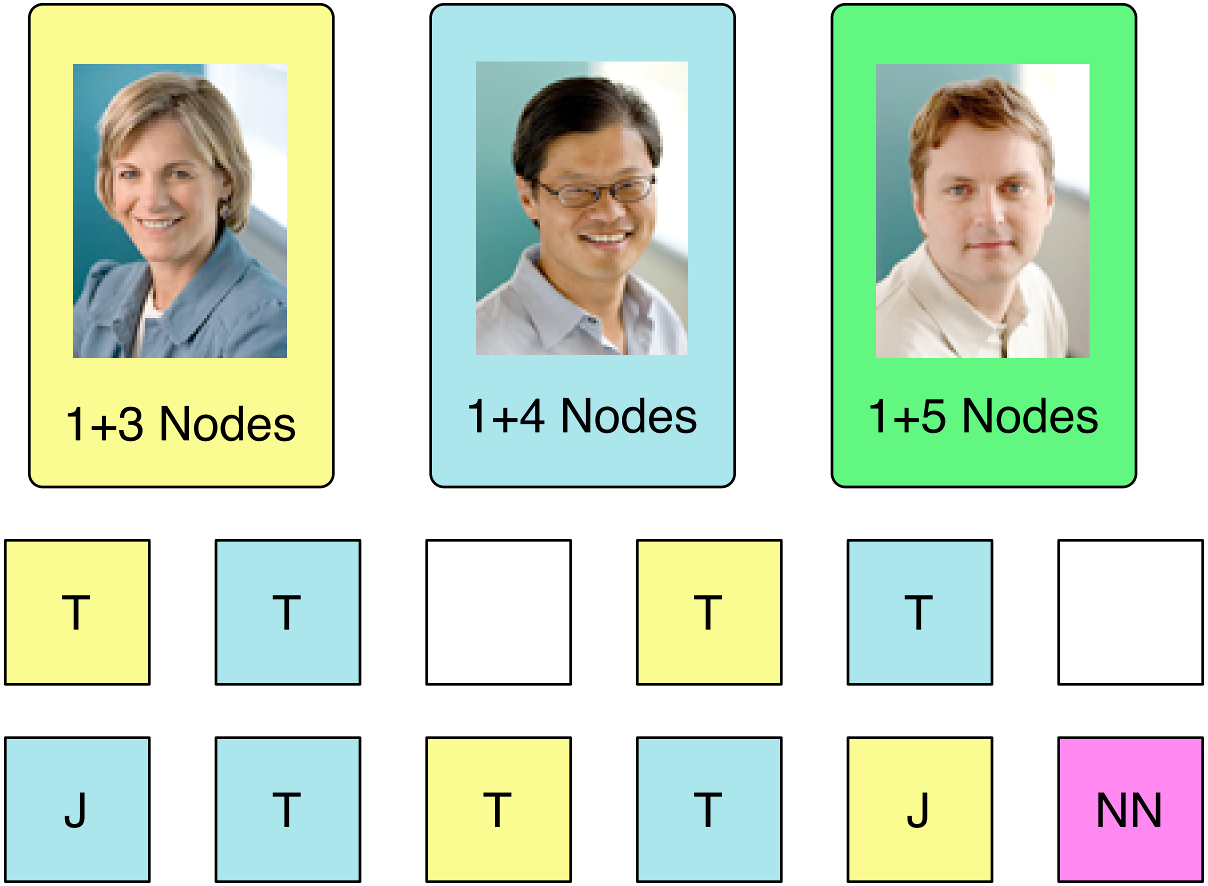
- Hadoop does not have an advanced scheduling system
  - MapReduce JobTracker manages one or more jobs running within a set of machines
  - Works well for “dedicated” applications, but does not work so well for shared resources
- Grid Services are intended to be a shared multi-user, multi-application environment
  - Need to combine Hadoop with an external queuing and scheduling system...

Hadoop On Demand (HoD)

- Wrapper around PBS commands
  - We use freely available Torque and Maui
- Big win: virtual private JobTracker clusters
  - Job isolation
  - Users create clusters of the size they need
  - Submit jobs to their private JT
- Big costs:
  - Lose data locality
  - Increased complexity
  - Lose a node for private JobTracker
  - Single reducer doesn't free unused nodes
    - ~ 30% efficiency lost!
- Looking at changing Hadoop scheduling
  - Task scheduling flexibility combined with node elasticity



HoD Job Scheduling



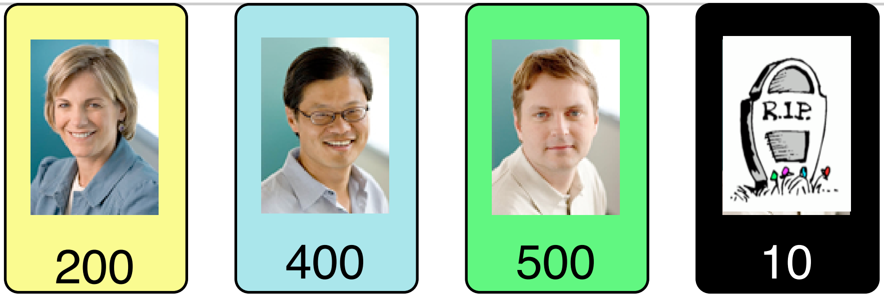
1+3 Nodes

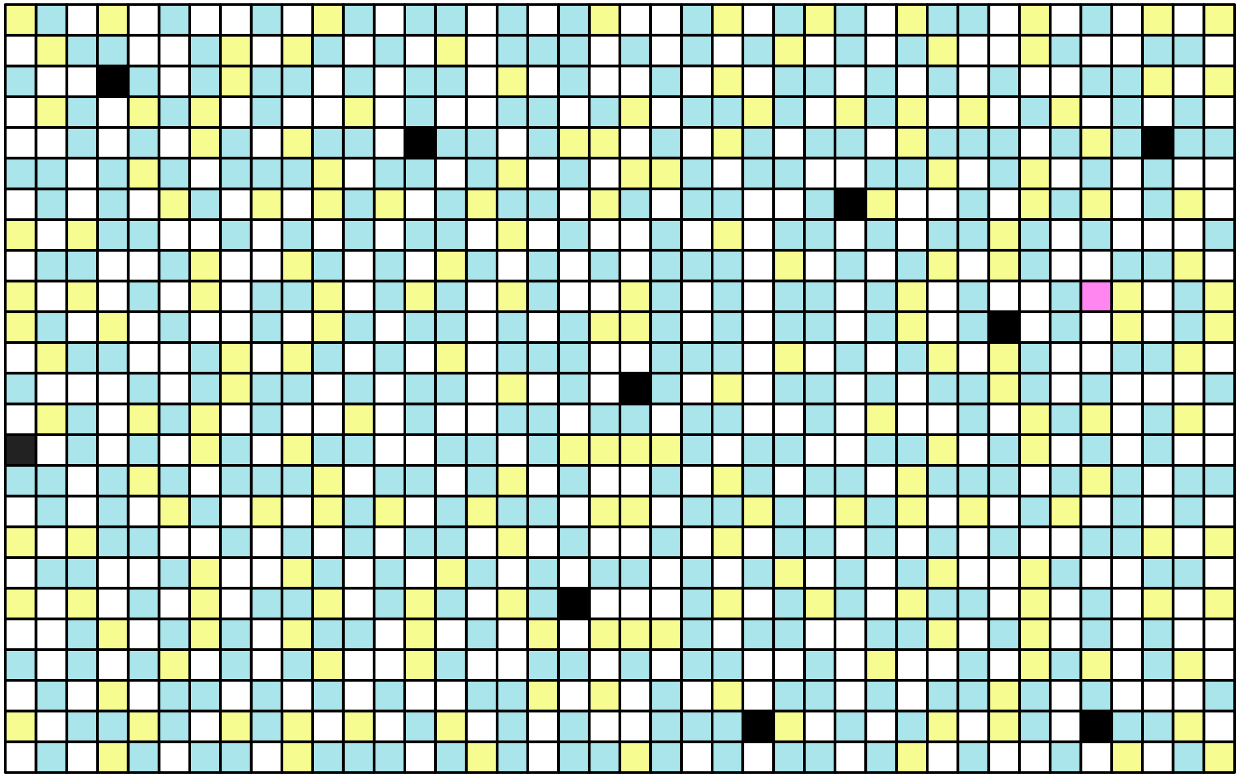
1+4 Nodes

1+5 Nodes

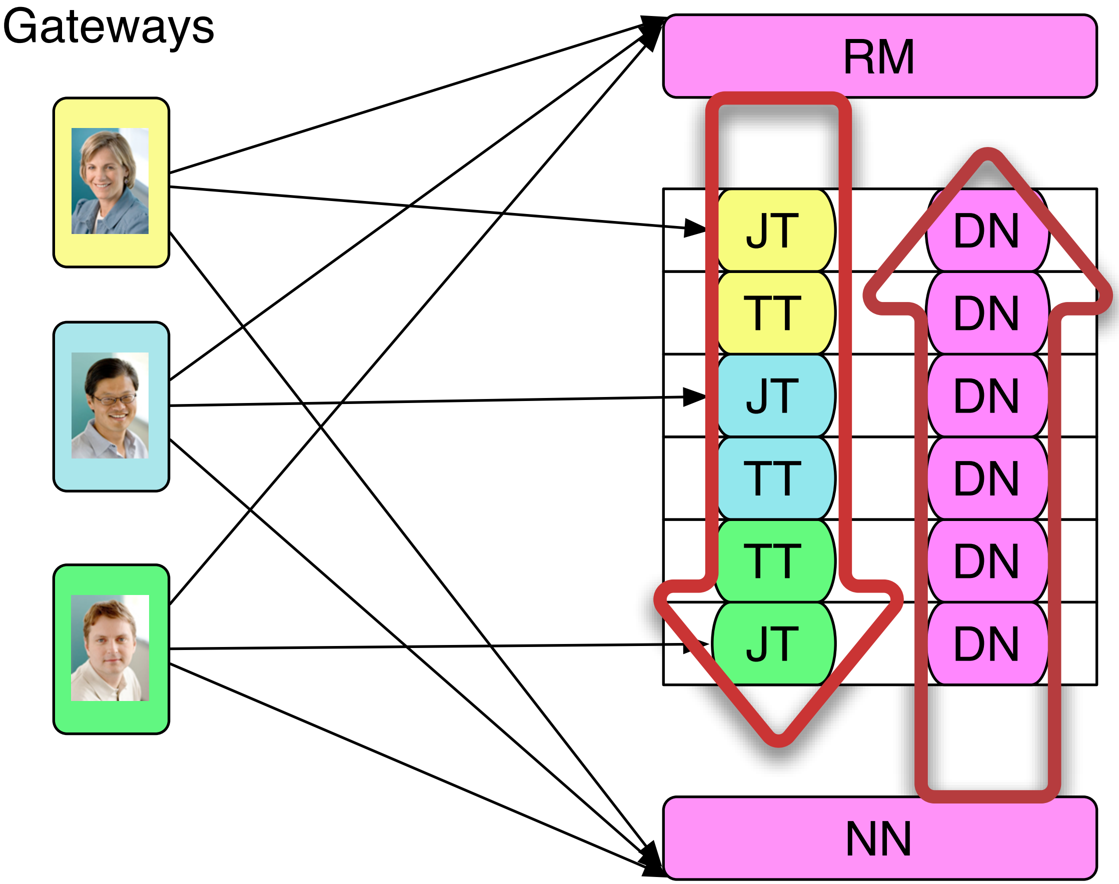
T	T		T	T	
J	T	T	T	J	NN

The Reality of a 1000 Node Grid

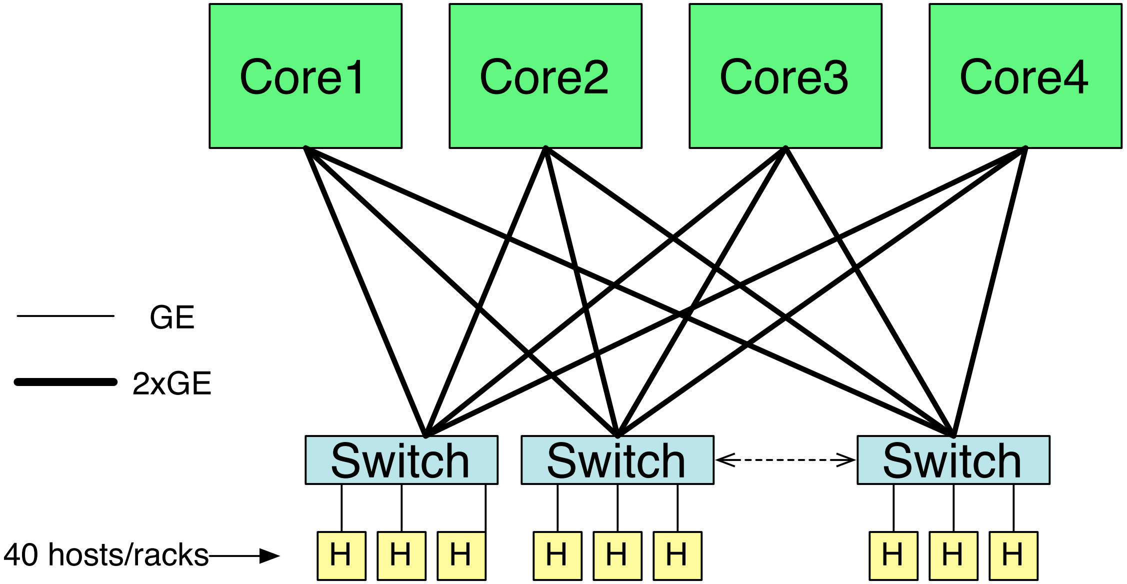




Putting It All Together



Network Configuration



Yahoo!’s Next Generation Grid Infrastructure

A Work In Progress

Background Information

- Internal deployments
  - Mostly Yahoo! proprietary technologies
- M45
  - Educational outreach grid
  - Non-Yahoo!'s using Yahoo! resources
    - Legal required us not to use **any** Y! technology!
- **Decision made to start from scratch!**
  - Hard to share best practices
  - Potential legal issues
  - Don't want to support two ways to do the same operation
- Internal grids converting to be completely OSS as possible
  - Custom glue code to deal with any Y!<-->OSS incompatibilities
    - user and group data

Naming and Provisioning Services

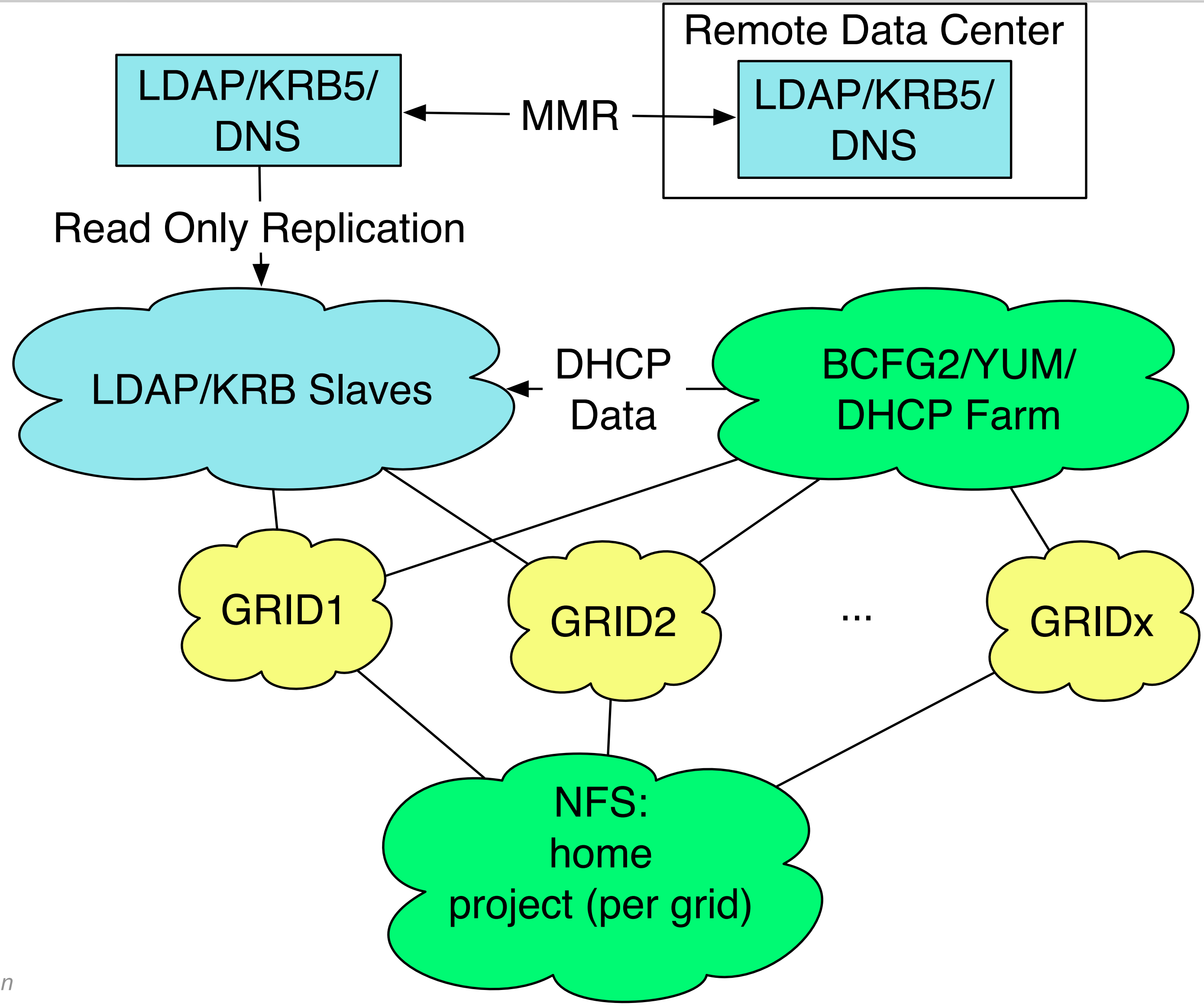
- Naming services
  - Kerberos for secure authentication
  - DNS for host resolution
  - LDAP for everything else
- ISC DHCP
  - Reads table information from LDAP
  - In pairs for redundancy
- Kickstart
  - We run RHEL 5.x
  - base image + bcfg2
- bcfg2
  - host customization
  - centralized configuration management

NFS for Multi-user Support

- NFS
  - Home Directories
  - Project Directories
    - Group shared data

- Grids with service level agreements (SLAs) shouldn’t use NFS !
  - Single point of failure
    - HA-NFS == $$$
  - Performance
  - Real data should be in HDFS

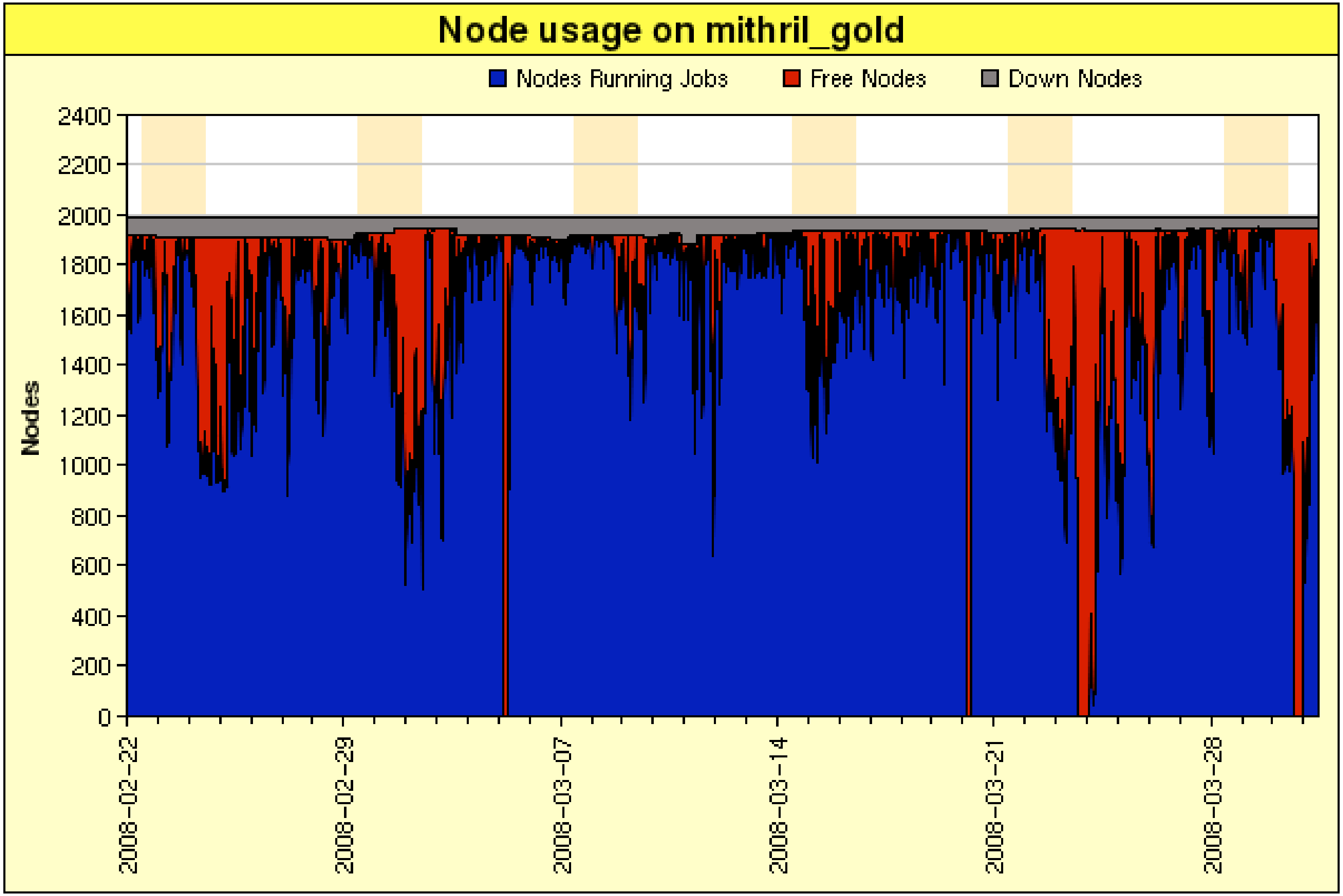
Big Picture


Remote Data Center
LDAP/KRB5/ DNS
MMR
LDAP/KRB5/ DNS
Read Only Replication
LDAP/KRB Slaves
DHCP Data
BCFG2/YUM/ DHCP Farm
GRID1
GRID2
...
GRIDx
NFS: home project (per grid)

Self Healing/Reporting

- Torque
  - Use the Torque node health mechanism to disable/fix ‘sick’ nodes
    - Great reduction in amount of support issues
    - Address problems in bulk
- Nagios
  - Usual stuff
  - Custom hooks into Torque
- Simon
  - Yahoo!’s distributed cluster and application monitoring tools
  - Similar to Ganglia
  - On the roadmap to be open sourced

Node Usage Report



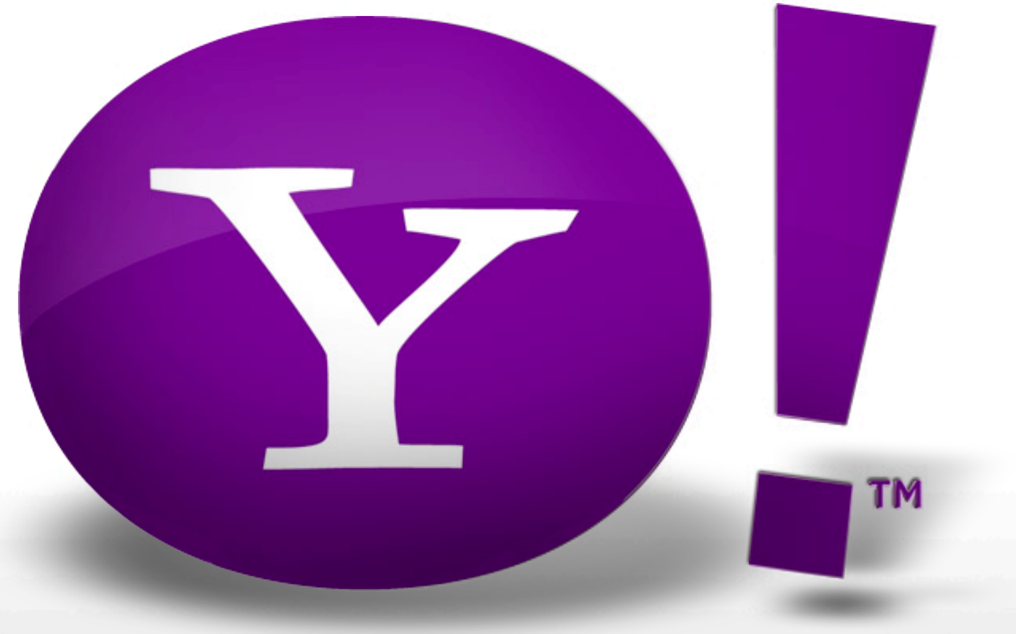
Ranges and Groups

- Range: group of hosts
  - example: @GRID == all grid hosts
  - custom tools to manipulate hosts based upon ranges:
    - ssh -r @GRID uptime
      - Report uptime on all of the hosts in @GRID

- Netgroup
  - Used to implement ranges
  - The most underrated naming service switch ever?
    - Cascaded!
    - Scalable!
    - Supported in lots of useful places!
      - PAM (e.g., _succeed_if on Linux)
      - NFS

Some Links

- Apache Hadoop
  - http://hadoop.apache.org/
- Yahoo! Hadoop Blog
  - http://developer.yahoo.com/blogs/hadoop/
- M45 Press Release
  - http://research.yahoo.com/node/1879
- Hadoop Summit and DISC Slides
  - http://research.yahoo.com/node/2104



YAHOO!