

# Linux Clusters

Greg Lindahl

CTO

Conservative Computers, Inc.

# What's a cluster?

- 3 kinds
  - High availability
  - High throughput (many identical operations)
  - High performance (parallel programs)
- Most clusters are a combination
- Most people don't explain what kind they're talking about
  - VMS people mean only a subset of these when they use the word "Cluster"

# Our Examples

- " google.com, search engine
- " slashdot.org, as a generic ecommerce site
- " Forecast Systems Lab, a Linux supercomputer, used to develop weather prediction programs
  - First Linux machine ever to win a traditional supercomputing procurement
  - \$15 million over 5 years
  - Alpha Linux, Myrinet interconnect, ya ya ya.

# High Availability

- „ I want to provide a service with "lots of 9's" e.g. 99.999% availability (5 9's, 5.3 minutes of downtime per year)
- „ Basic model: I have 2 identical providers of service; if one isn't available, I use the other.
  - "fail over": user doesn't have to do anything to switch
  - "fall over": users have to do something

# High Availability 2

- „ Most "clustering" is just HA clustering (e.g. NT Wolfpack)
- „ Hard to do transparently with many kinds of services
  - NFS uses the disk inode #'s all over the place, so NFS failover has to present exactly the same inodes for all files
  - "ip takeover" usually can't preserve existing TCP streams, so clients have to reconnect

# High Throughput

- Example operations:
  - I have a program that matches a gene sequence against a database. I want to run dozens of searches against a variety of databases.
  - I have a bunch of users who want to access my website, which has a variety of static and active content.
- Basic model: route the request to one of  $N$  systems that can do the work

# High Performance

- „ I have a parallel program that I want to run fast.
- „ One common programming model: MPI
- „ Clusters can be very large: 100 to 500 nodes are common, 2,000 to 5,000 not so uncommon
- „ Usually fairly general purpose machines with many users, who have to share the machine

# But...

- „ Most systems want more than 1 kind of clustering
- „ Most tools implement more than one kind, and they can't necessarily be combined
- „ The devil is often in the details.

# Details, Details

## " Slashdot

- I need that page NOW luuserz!
- Relatively small system (less than a dozen servers, 1 database server)
- 3133t software goes \*boom\* on a regular basis

## " FSL

- I want my to run within a couple of hours
- 410 nodes
- Nodes crash fairly rarely (a few per month)

# google.com

- Google is even more interesting; it has many kinds of clustering in one cluster:
  - Front ends to do searching against a database (1/2 second response time is their goal)
  - Spider to build next month's database
  - cache of old webpages
- All built using 1U systems with a couple of IDE disks each. 3,000 nodes, 5,000 cpus...

# Cluster Tools

- <http://linux-ha.org/> -aimed at high availability
  - Heartbeat component
  - "after something dies, future incoming connections get handled by someone else"
- <http://LinuxVirtualServer.org/>
  - redirects all incoming connections to one of N servers
  - combination of high availability and high throughput

# Cluster Tools 2

- drbd Distributed Redundant Block Device
  - Allows 2 systems to share a copy of a block device (disk) using a private network
  - Master/slave, only one has read/write at a time
  - When one fails, the other has to fsck before starting to use the filesystem, so use a journaling fs
  - In combination with heartbeat and ip takeover, can provide a redundant NFS server

# Cluster Tools 3

## " MOSIX

- Extensive kernel hacking
- Attempts to transparently migrate processes among several machines
- Useful for load balancing (high throughput), but generally provides lower availability

# Cluster Tools 4

## • Condor

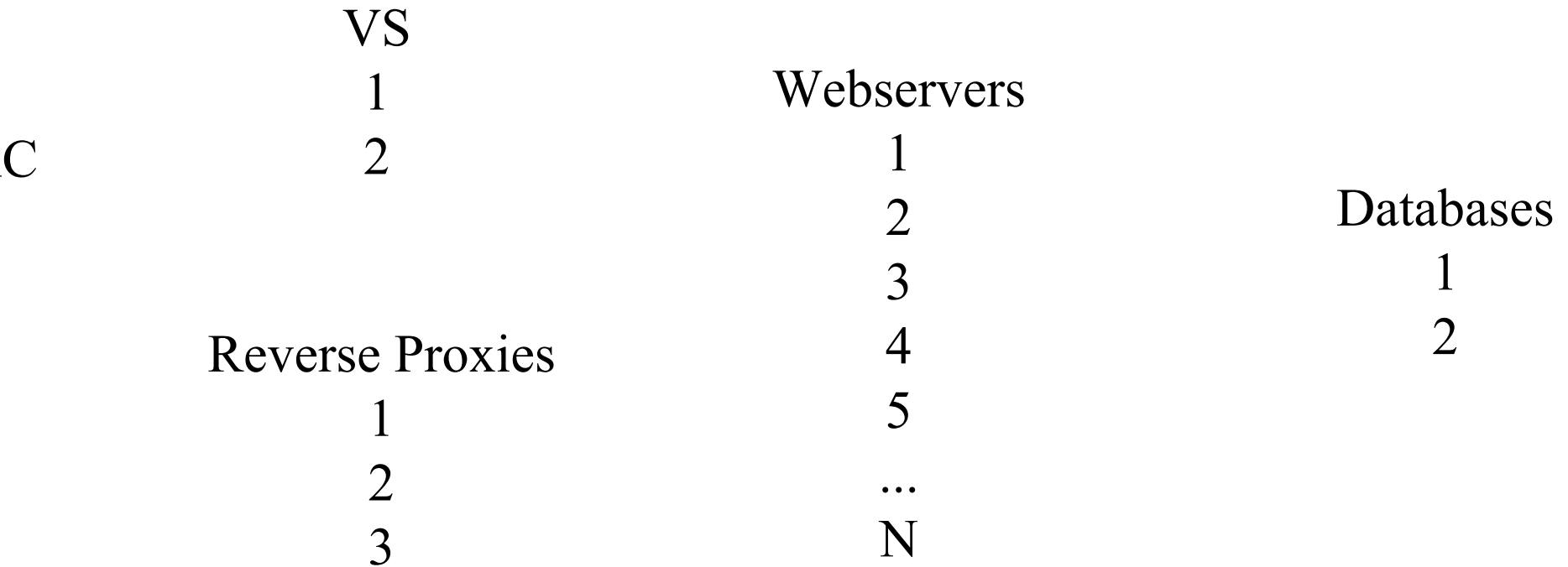
- "batch queue system": hand it 1,000 jobs and 100 idle workstations, and it does the right thing
- Also has checkpoint/restart (mostly hacked into glibc etc.)
- Has some higher availability features: a node crashed running job X, let's rerun that job on a different node

# Cluster Tools 5

- Traditional Queue Systems
  - Run either lots of serial or parallel jobs
  - Complicated scheduling algorithms

# Putting it together: Slashdot (or any ecommerce site)

• N tier architecture



# Putting it together: FSL

- „ Uses a traditional queue system
- „ Custom heartbeat system ("reliability daemon") written in perl
- „ When a node fails, it asks the queue system to rerun any batch job using the node.
- „ 99.9% reliability for this part of the system

# More Cluster Tools

- „ "Cluster Filesystems"
- „ Capabilities depend on what you want from your "cluster"
- „ GFS: all things to all people
- „ drbd mentioned earlier
- „ Intermezzo
- „ Lots of commercial products in this space: IBM GPFS, SGI CXFS, Adic CVFS

# More Cluster Tools 2

- „ System administration
  - I have a bunch of boxes now, how do I administer them?
- „ Major approaches
  - Treat it like any other group of boxes, e.g. use cfengine
  - Maintain them in lock step (IBM SP, FSL)
  - Use a strict master/worker system

# More Cluster Tools 3

## " Scyld Beowulf

- Master has a disk, normal looking Linux install
- Workers all boot diskless, get a limited OS from the master
- Workers can still have a local disk for local data
- Traditional batch queue on the master, or you can "brsh foo" to run a command on a worker

Questions?