LINUX HA IN A HIGH PERFORMANCE ENVIRONMENT

Eric Blau

Overview

- What is Linux HA?
- Uses
- Terminology
- Versions of Linux HA
- Configuration
- Resources
- Split Brain
- Tekelec’s use of Linux HA

What is Linux HA?

- Cluster infrastructure framework
  - Communication and membership services
  - Discovery of peer processes on other servers in the cluster
  - Often referred to as “heartbeat” – status messages sent between cluster members
- Resource management
  - Starting and stopping of services to keep services available
  - Often referred to as the “Cluster Resource Manager (CRM)” or “Pacemaker”

Why use Linux HA?

- Protect from hardware failures, unexpected software errors
- Load balancing
- Maintain service during maintenance windows
- Rule of thumb: HA adds “1 nine” to uptime. 99.9% uptime -> 99.99% uptime

Linux HA Terminology

- Node
  - A single server running HA.
- Cluster
  - One or more nodes running HA managing a common set of resources.
- Resource
  - A service managed by HA.
  - Linux HA starts and stops resources.
- Score
  - Numeric value expressing HA preference. A higher HA score means that a node is “more preferred” to run a given resource.
- Location constraint
  - Restriction, using score values, for specifying which node a given resource should be started on.
  - Location constraints can be used to express resource dependencies (e.g., an IP should only be added on the node that has mastership of the database).
- Stickiness
  - How resistant, using score values, a resource is to moving away from the current node it is started on.

Versions of Linux HA

- Version 1.x – linux-ha.org
  - Supports only simple 2-node clusters
  - No resource monitoring
- Version 2.x – linux-ha.org
  - Supports multiple nodes in a cluster (up to 16 nodes tests)
  - Resource monitoring to detect failures and restart or move them to other servers
- Pacemaker – clusterlabs.org
  - Project reorganization with the CRM split out from heartbeat and the ability to run on the alternative OpenAIS cluster infrastructure

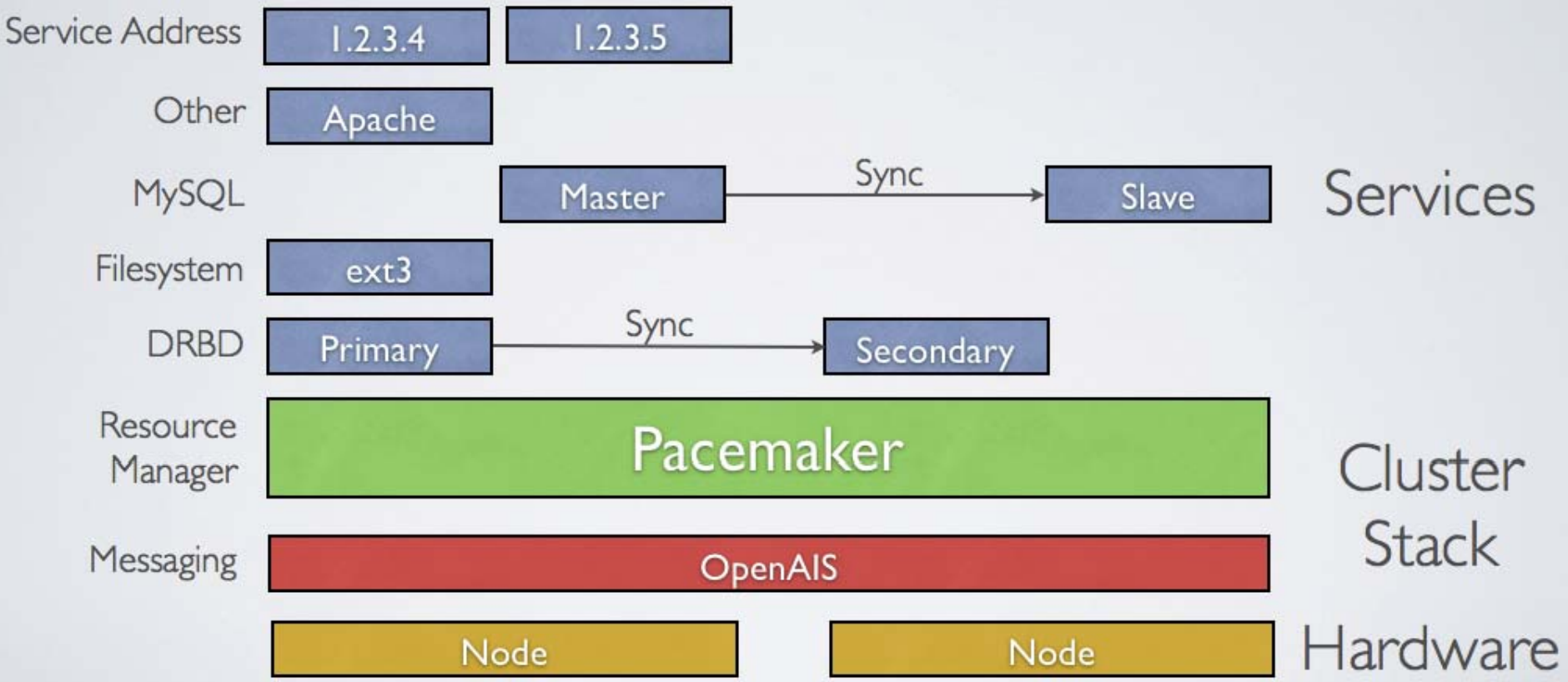
Nodes and Clusters

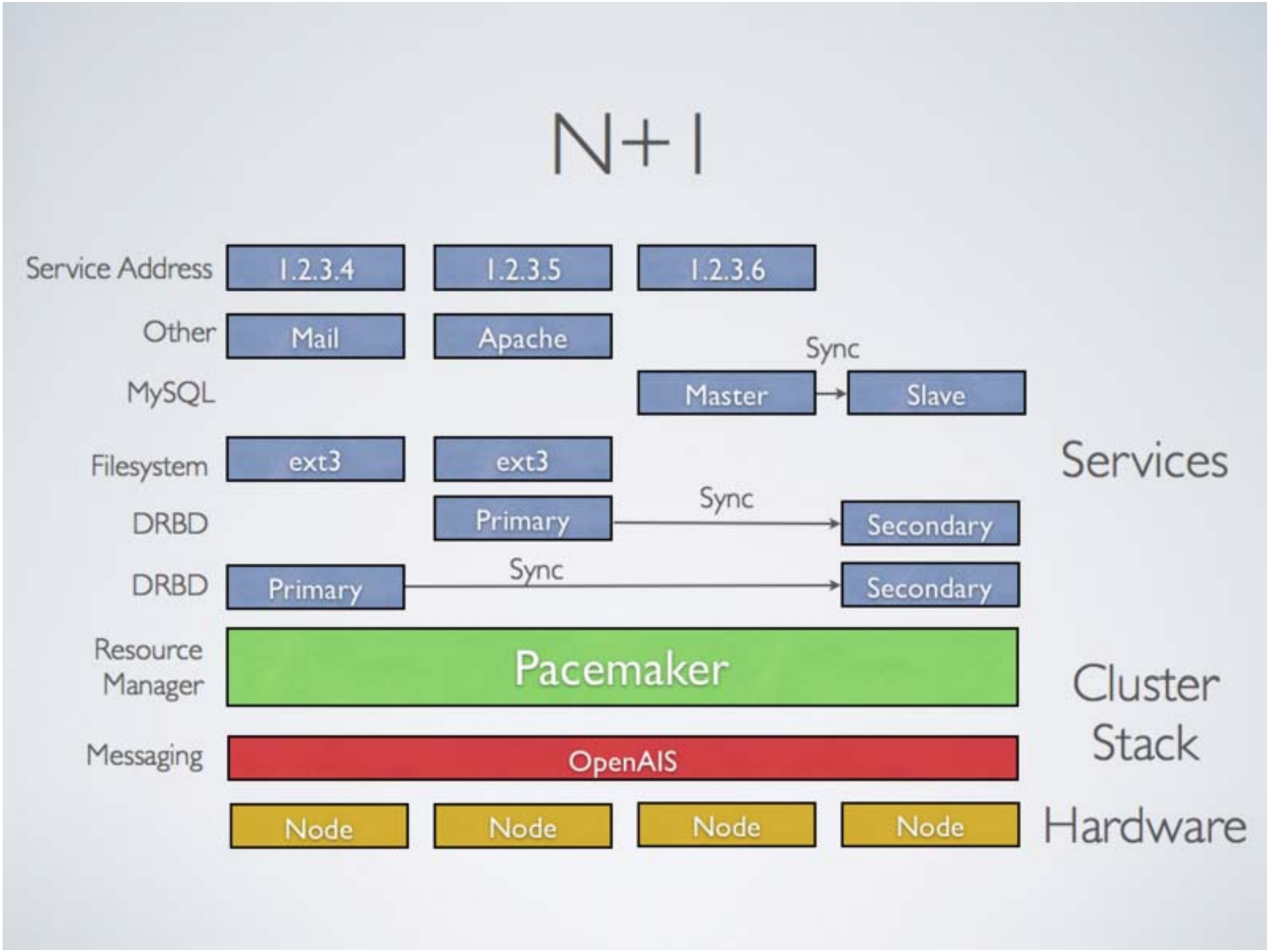
- Configuration file specifies the nodes that are in a given cluster
- Communication path for nodes to communicate
  - UDP (port 694 by default) using unicast, broadcast and multicast IP addresses
  - Serial port communication is also supported
- Other cluster configuration options

Cluster configuration options

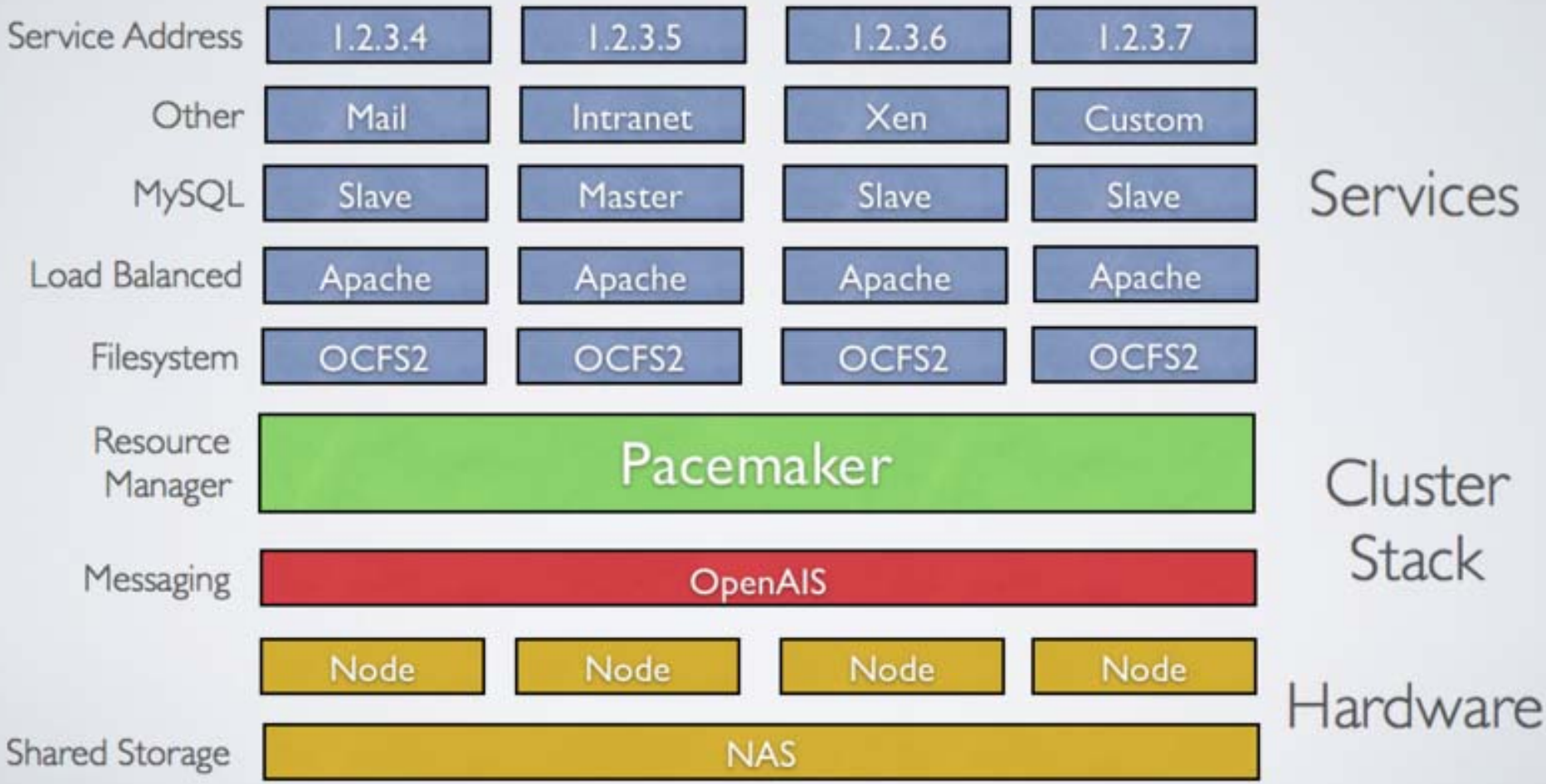
- Active/passive – one server providing service with one other server ready to take over if needed
- N+1 – many servers providing service with one backup server capable of taking over if any server fails
- N-to-N – all servers providing service with any server capable of handling another server's load in the event of a failure

ACTIVE/PASSIVE




N+1
Service Address
1.2.3.4
1.2.3.5
1.2.3.6
Other
Mail
Apache
Sync
MySQL
Master
Slave
Filesystem
ext3
ext3
Services
Sync
DRBD
Primary
Secondary
Sync
DRBD
Primary
Secondary
Resource Manager
Pacemaker
Cluster Stack
Messaging
OpenAIS
Node
Node
Node
Node
Hardware

N-TO-N


Service Address
1.2.3.4
1.2.3.5
1.2.3.6
1.2.3.7
Other
Mail
Intranet
Xen
Custom
MySQL
Slave
Master
Slave
Slave
Services
Load Balanced
Apache
Apache
Apache
Apache
Filesystem
OCFS2
OCFS2
OCFS2
OCFS2
Resource Manager
Pacemaker
Cluster Stack
Messaging
OpenAIS
Node
Node
Node
Node
Hardware
Shared Storage
NAS

Resources

- Resources are managed using scripts similar to LSB compliant init scripts
- For Linux HA v1, the resource script must support 3 operations:
  - start – initiate or gain control of the resource
  - stop – terminate or relinquish control of the resource
  - status – report whether the resource is started or stopped
- For v2 configurations, the script must support an additional “monitor” operation

Applications for Linux HA

- The following types of applications are typical:
  - Database servers
  - ERP applications
  - Web servers
  - LVS director (load balancer) servers
  - Mail servers
  - Firewalls
  - File servers
  - DNS servers
  - DHCP servers
  - Proxy Caching servers
  - Custom applications

Split Brain

- Situation that occurs when multiple nodes in a cluster believe the other server is dead.
- Creates an active-active condition where a single resource is started on multiple nodes
- Should be avoided at all costs, especially if shared storage is involved – data corruption!
- Redundant communication paths help prevent split brain

Split Brain (cont.)

- Can be mitigated/prevented using several strategies:
  - Quorum – with an odd number of nodes, only the node with a quorum can become active
  - Fencing – protect again nodes of unknown status from accessing/running resources
  - STONITH (“shoot the other node in the head”) – specific type of fencing
  - Scoring – provide a location constraint score to allow HA to decide which server to become active when split brain is healed

Tekelec's problem

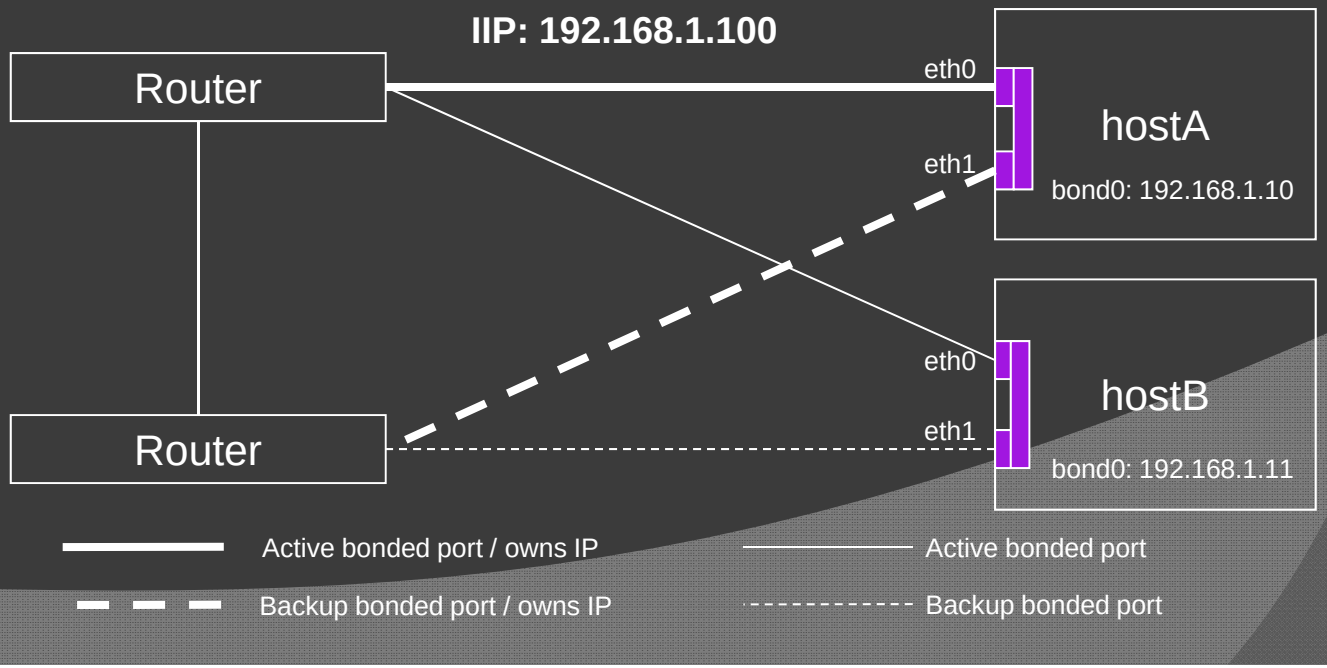
- Proprietary shared memory database and middleware layer
- Need for high performance HA solution to ensure 99.999% availability
- HA needed to manage:
  - Database replication master/slave roles
  - IP address(es) for database clients to access
  - Process awareness of activity (active/standby)

Tekelec's solution

- Linux HA version 2.x in an active/passive configuration
- One or more IP addresses and database resource configured
- Uses cluster messaging to communicate application status
  - Alarms used to trigger location constraints that influence HA activity
  - Heartbeat interval set low to detect failures more quickly (100-250 milliseconds)
  - Database resource communicates HA state change to other processes registered for notification

Tekelec's solution (cont.)

- To avoid split brain, port bonding in an active/backup configuration is used with redundant switches.
  - No single point of failure



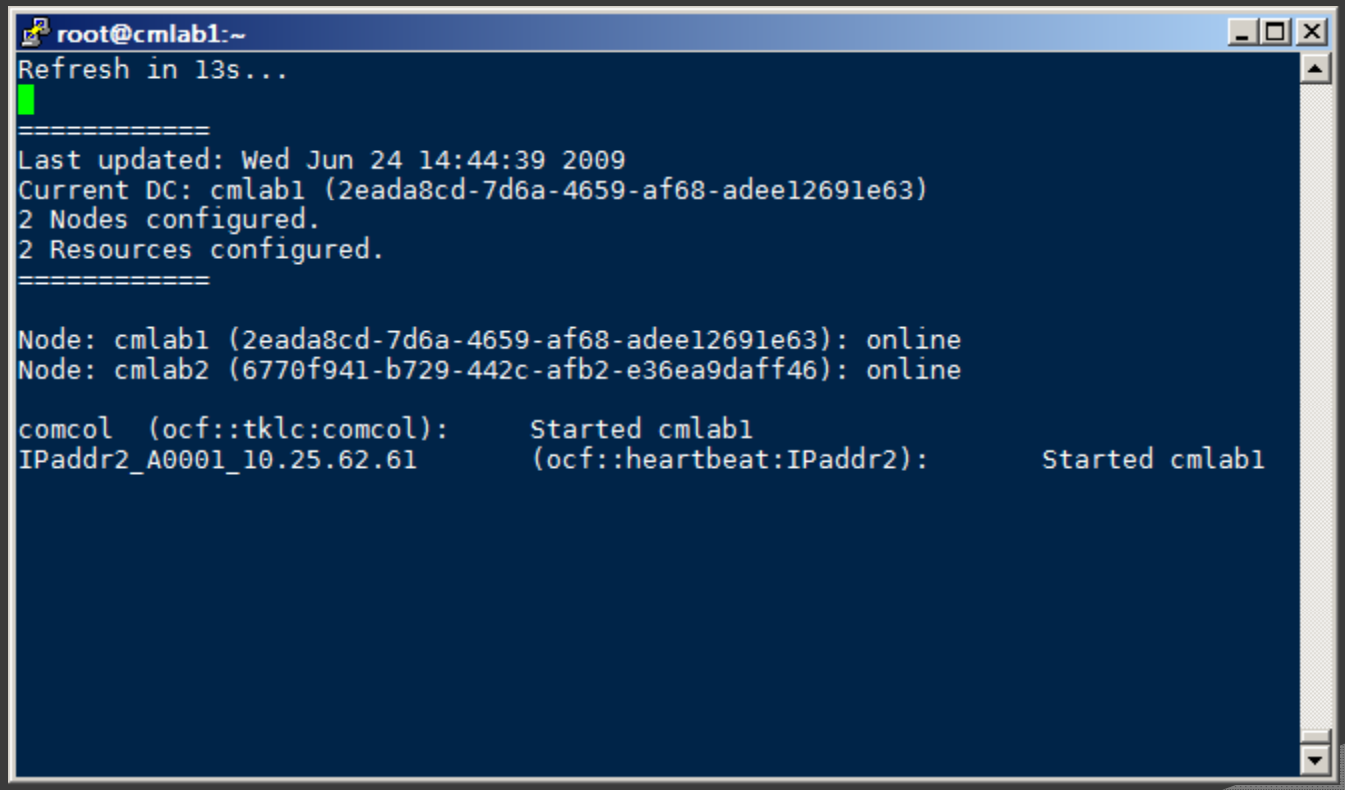
Limitations of Linux HA v1

- Several significant limitations with previous Linux HA version 1 configuration
  - No split brain recovery. After a split brain, the server that becomes active is essentially random. If the wrong choice is made, data will be lost.
  - No resource monitoring. Processes may be stopped on a given node, but Linux HA will never find out about it. Linux HA may decide to switch over to a node where the database is not sane (corrupted or auditing).
  - Only supports 2 node active/standby clusters.
  - Rudimentary support for resource dependencies or location constraints.
- Bottom line: Linux HA version 1 configurations are not robust enough and do not provide sufficient guarantees to ensure database integrity.

HA Scoring

- HA scoring controls on which server Linux HA makes database active vs. standby.
- Linux HA refers to this score as a “resource location constraint”
  - The higher the score, the more preferred the resource is to run (e.g., make database active) on a server
  - The lower the score, the less preferred the resource is to run on a server
- The DB state and current set of raised alarms are inputs to the HA score
- Each server publishes its own score to Linux HA
- When one server’s score published to Linux HA is higher than another, a switchover is initiated

Example Linux HA status (crm_mon command)



Observations

- Linux HA works quite well in high performance environments
- Cluster configuration is extremely flexible – many different HA setups are possible
- Switchovers can generally be performed in < 1 second
- Version 2 configurations can be difficult to set up. The XML-based configuration is complex.

Questions?

- Any questions?
- www.linux-ha.org
- clusterlabs.org
- http://clusterlabs.org/mediawiki/images/f/fb/Configuration_Explained.pdf