

Clusters

Григорий Хазанкин , Данила Ковалёв

Что?

1. Распределенная компьютерная система - отказоустойчивая виртуально единая система компьютерных ресурсов, для которой характерна прозрачная для пользователей интеллектуальная агрегация ресурсов.
2. Консолидация компьютерных ресурсов - объединение аппаратного обеспечения в компьютерную систему
3. Виртуализация компьютерных ресурсов – изолированное использование «части» компьютерных ресурсов

Кратко о главном

Зачем ?

- Увеличение скорости вычислений
- Балансировка нагрузки
- Обеспечение надежности

Как?

- Volunteer Computing (Grid)
-
- GNU/Linux (MOSIX, Linux-HA),
Windows (CCS)
-
- Supercomputer

Grid & Supercomputer

Grid:

- Узлы разнородны (Win, UNIX) подключены к сети (локальной или глобальной) при помощи обычных протоколов (Ethernet)
- Узел ненадежен и может отсоединиться в любой момент
- Такая нестабильность компенсируется большим количеством узлов

Supercomputer:

- Узлы однородны, подключены к высокоскоростной коммутируемой компьютерной сети (InfiniBand, FC)

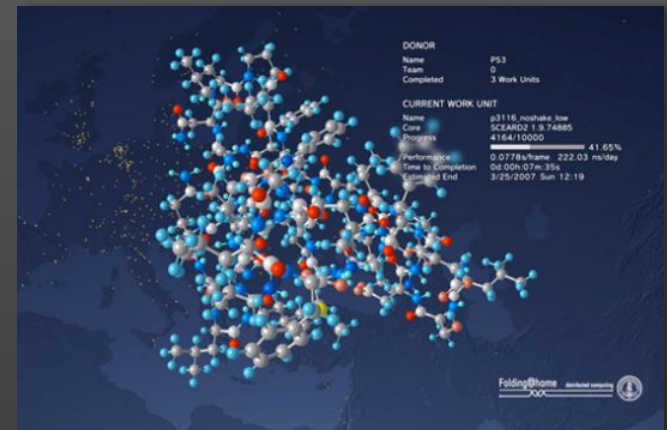
Grid & Supercomputer

Пример Grid:

Моделирование свертывания белка в Стэнфорде

FOLDING@HOME

- 222 000 CPU & 23 000 GPU
- Intel, AMD, Cell, AMD/NVIDIA GPU
- 7 PetaFLOPS (2011)
- Архитектура «клиент-сервер»



Grid & Supercomputer

Пример Supercomputer:

Оборонный научно-технический университет

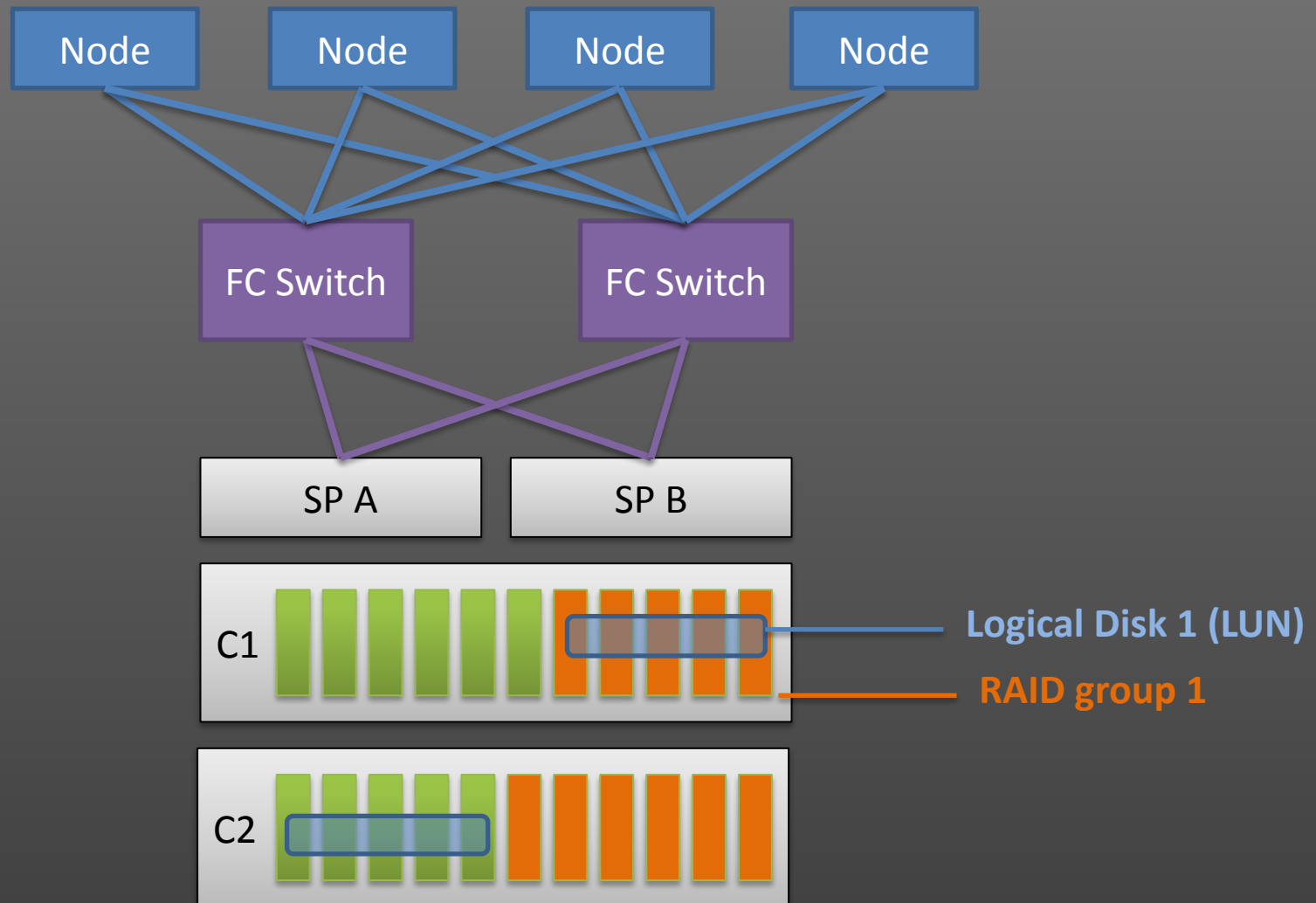
НОАК

天河二號, Tiānhé-2

- 33,86 PetaFLOPS
 - СОСТОИТ ИЗ
- 16 тысяч узлов
(2 процессора Intel Xeon E5-2692 на архитектуре Ivy Bridge с 12 ядрами (частота 2,2 ГГц)
и 3 сопроцессора Intel Xeon Phi 31S1P



Аппаратная архитектура. Supercomputer



Computational clusters

Использование: ресурсоемкие
вычисления

MPI (MPICH, Open MPI)

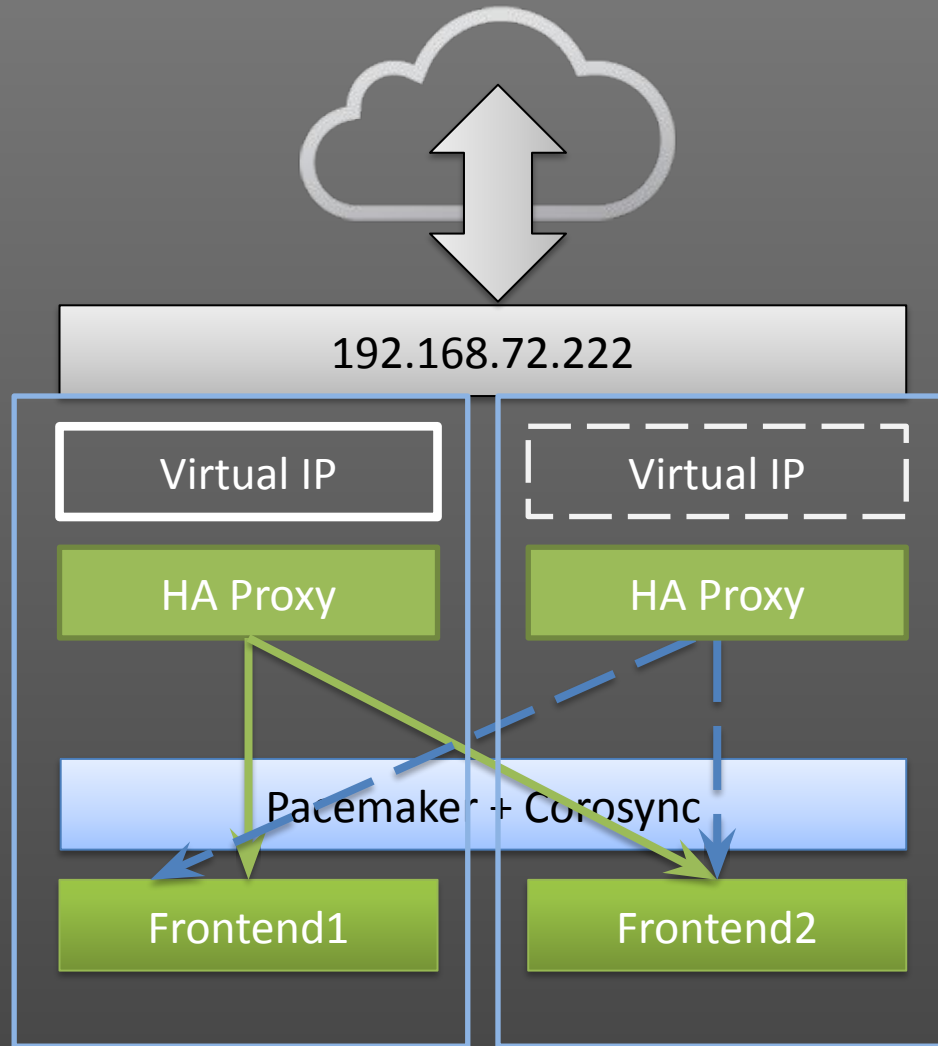
Hadoop

LB & HA clusters

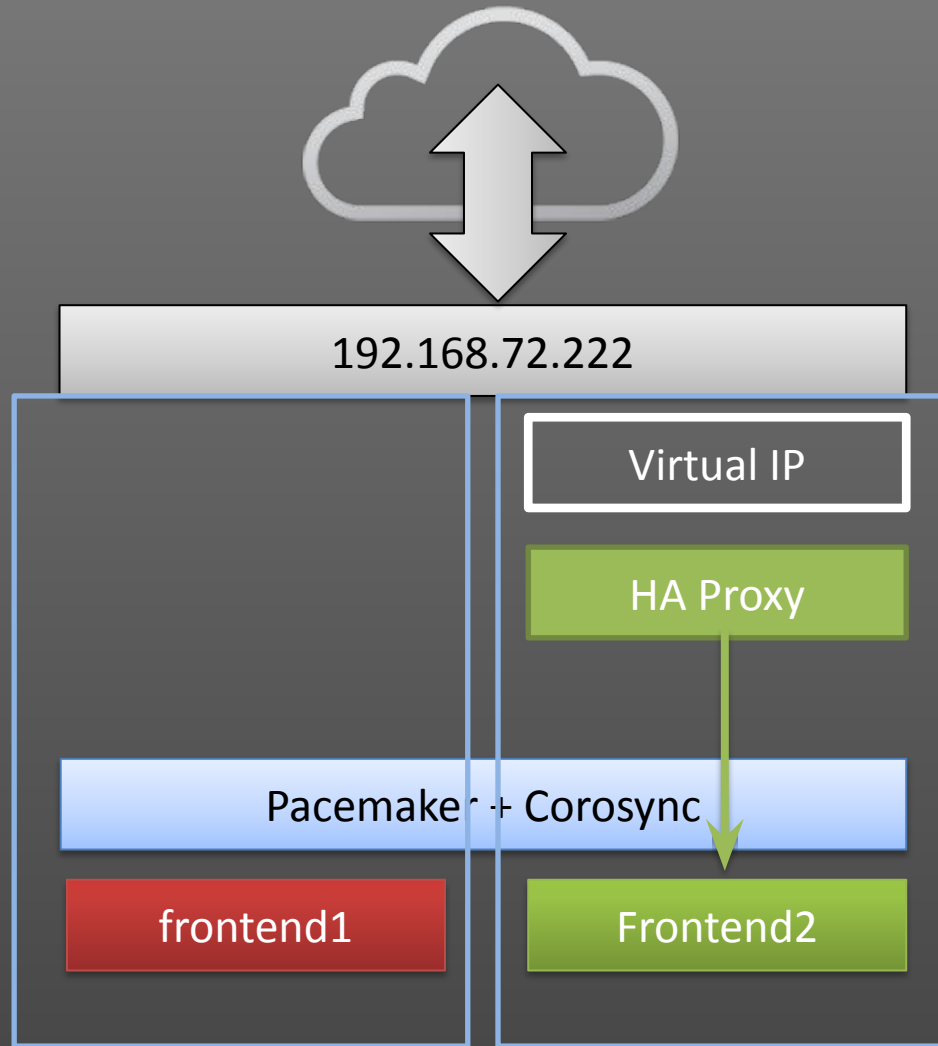
Задача кластеров высокой доступности
High-availability clusters / failover clusters -
организовать надежный доступ к сервисам путем
обеспечения отказоустойчивости.

Задача кластеров балансировки
Load balancing clusters –
распределение нагрузки между нодами
(вершинами) кластера для повышения
производительности.

LB & HA clusters. Example (Normal)



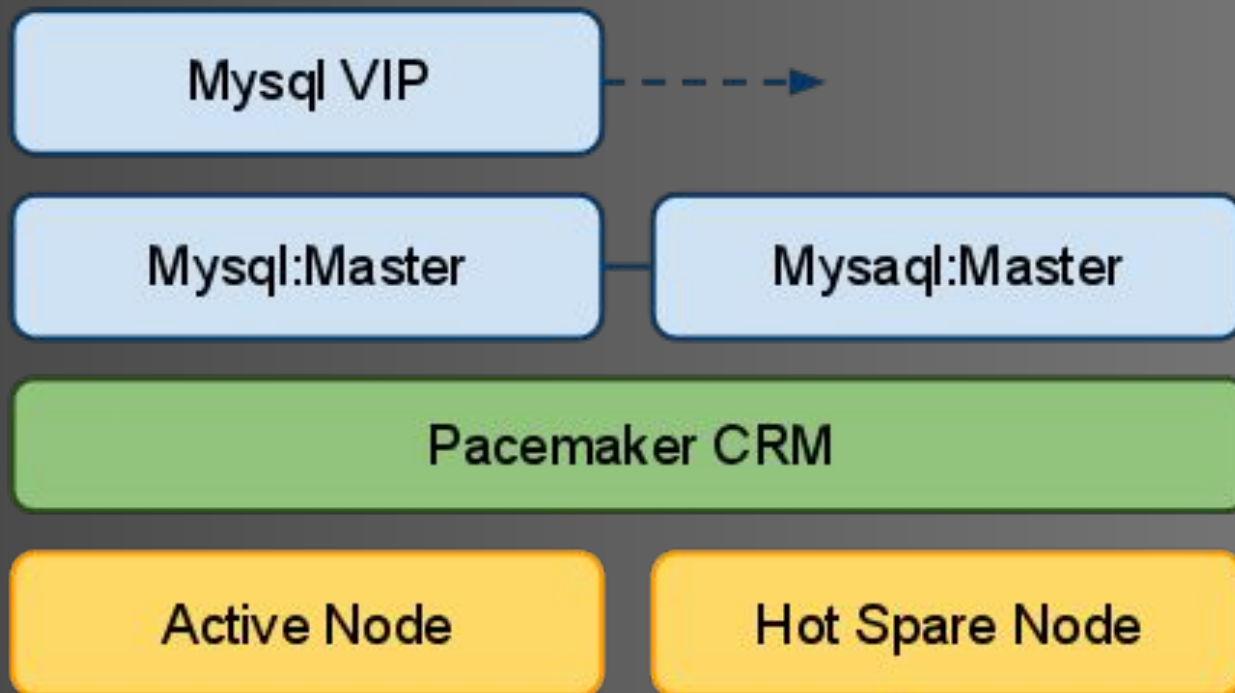
LB & HA clusters. Example (Fail)



Принцип работы НА. Ресурсы

- Все, что может быть заскриптовано, то может быть ресурсом. Все, что требуется от скрипта, это выполнять 3 действия: start, stop и monitor. Скрипты должны соответствовать LSB (Linux Standard Base) или OCF (Open Cluster Framework). Ресурс может представлять из себя: ip-адрес, демон с определенной конфигурацией, блочное устройство, файловую систему и т.д.
- Ресурс соответственно должен уметь хранить свое состояние таким образом, чтобы его можно было с минимальными потерями перенести на другую ноду

Принцип работы HA. Ресурсы

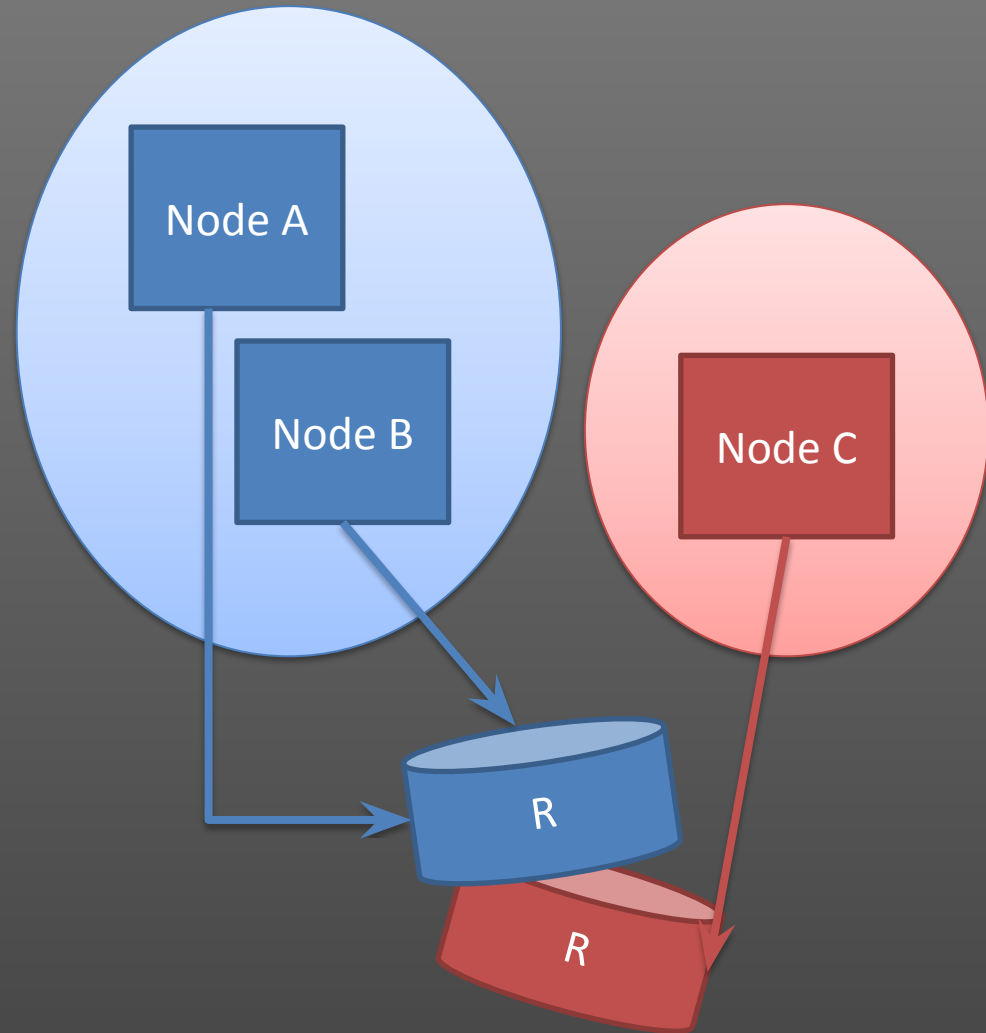


Split-Brain

- Кластер: 3 ноды А, В, С. Ресурс R находится на ноде А
- Теряется связь между А и остальными.
- В и С не знают, работает ли А (мертвый\живой).
- Поэтому они хотят перехватить управление ресурсом, чтобы пользователь не заметил возможной смерти А.
- Однако А жив (просто у него нет связи с остальными) и продолжает управлять ресурсом R.
- Две изолированных группы нод управляют одним ресурсом R одновременно, что может привести к повреждению ресурса.
- Возникает ситуация Split-Brain (“помутнение разума”)

Split-Brain

Иллюстрация
сломанного
ресурса

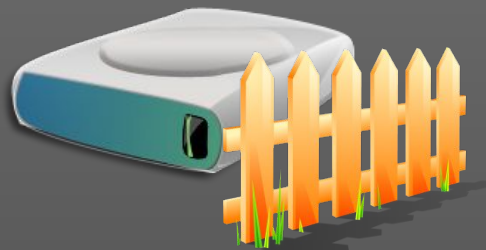


Fencing

Решение проблемы со Split-Brain: отрезать «выпавшей» ноде доступ к ресурсу и самим управлять им

Два типа fencing:

- Resource-fencing
- Node-fencing



Fencing

Node fencing:

- Киллер на службе у кластера (тупо гасит «плохую» ноду)
- stonithd daemon
- STONITH plugin

STONITH devices:

- UPS
- PDU
- Blade power control device
- Lights-out devices
- Testing devices



Resource fencing

Resource fencing:

Кластер может иметь возможность контролировать определенные устройства (например, дисковые массивы, свитчи и т. д.) для того, чтобы запретить к ним доступ для «плохих» нод.

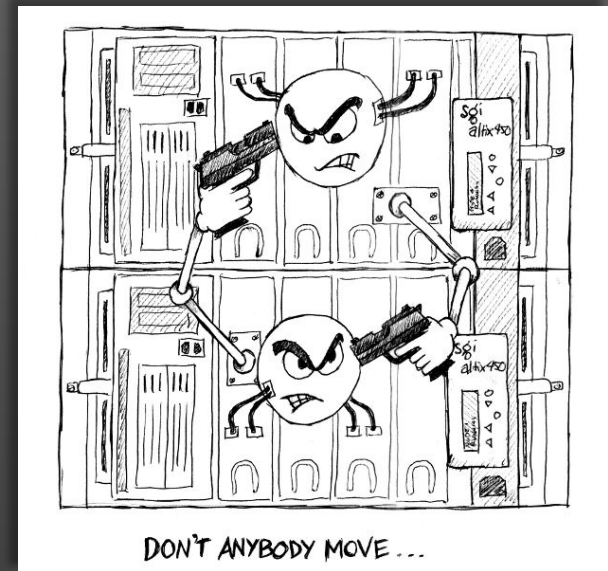


Quorum

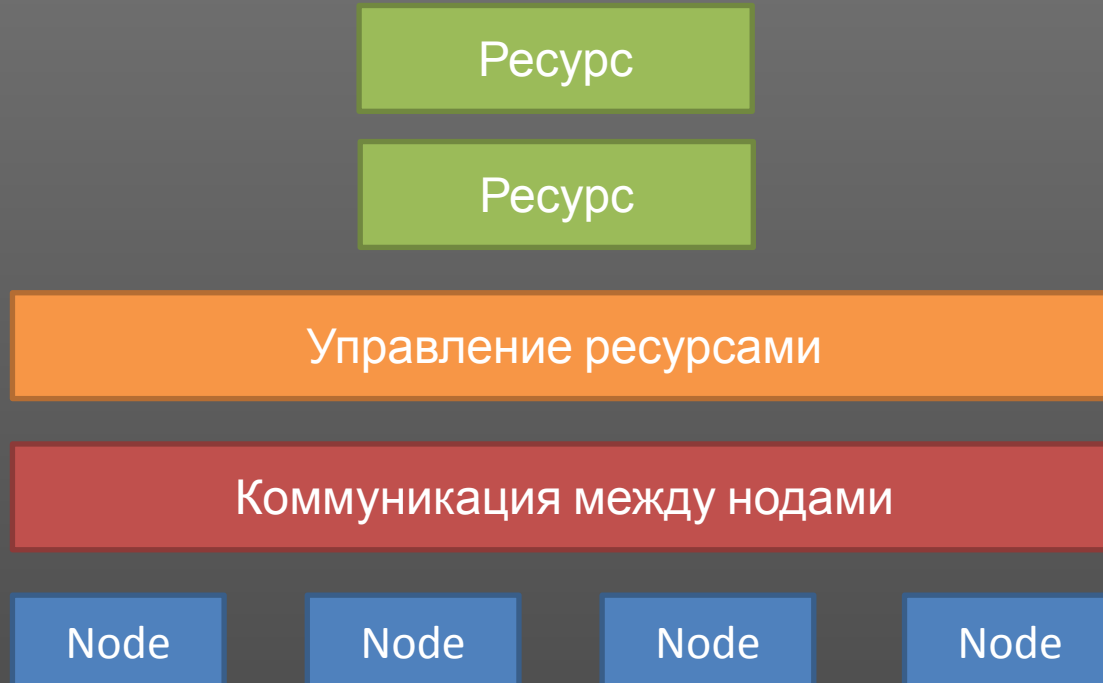
При Split-Brain каждая независимая группа нод будет пытаться сделать STONITH другой группе нод. Т. о. ничего работать не будет (все умрут). Поэтому нужен некий механизм арбитража, для выяснения, кого именно «грохнуть».

Логика кворума проста: если я могу достучаться до более, чем $N/2$ нод, то нужно убить другую группу нод.

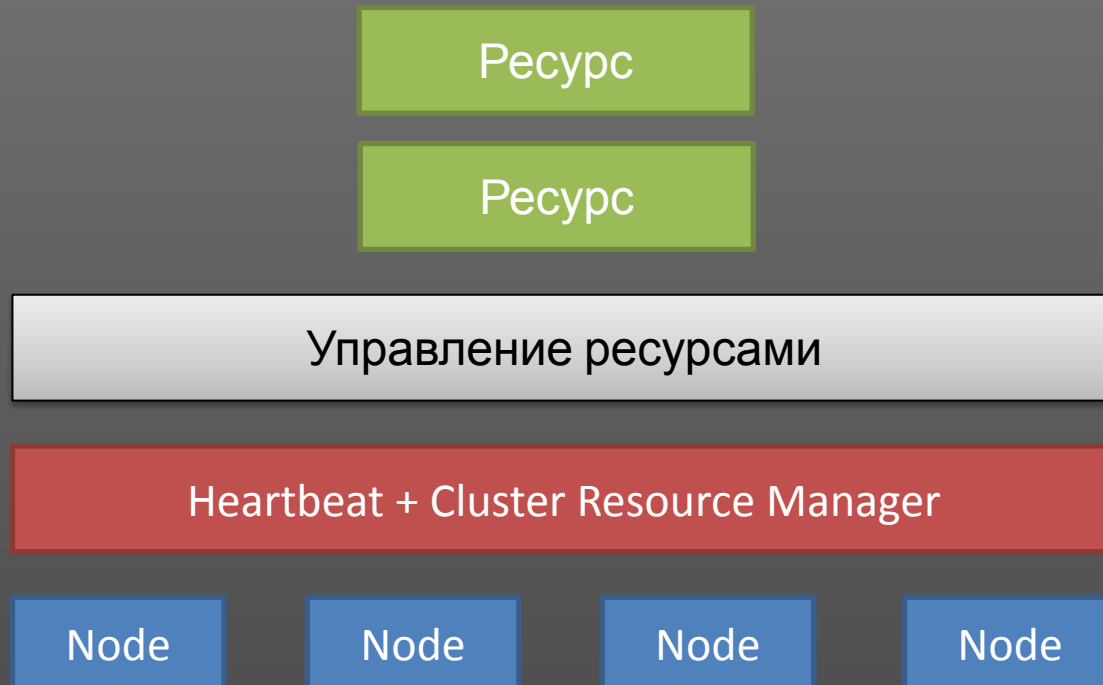
Поэтому: выключайте кворум и STONITH на 2-node-cluster, иначе все повиснет.



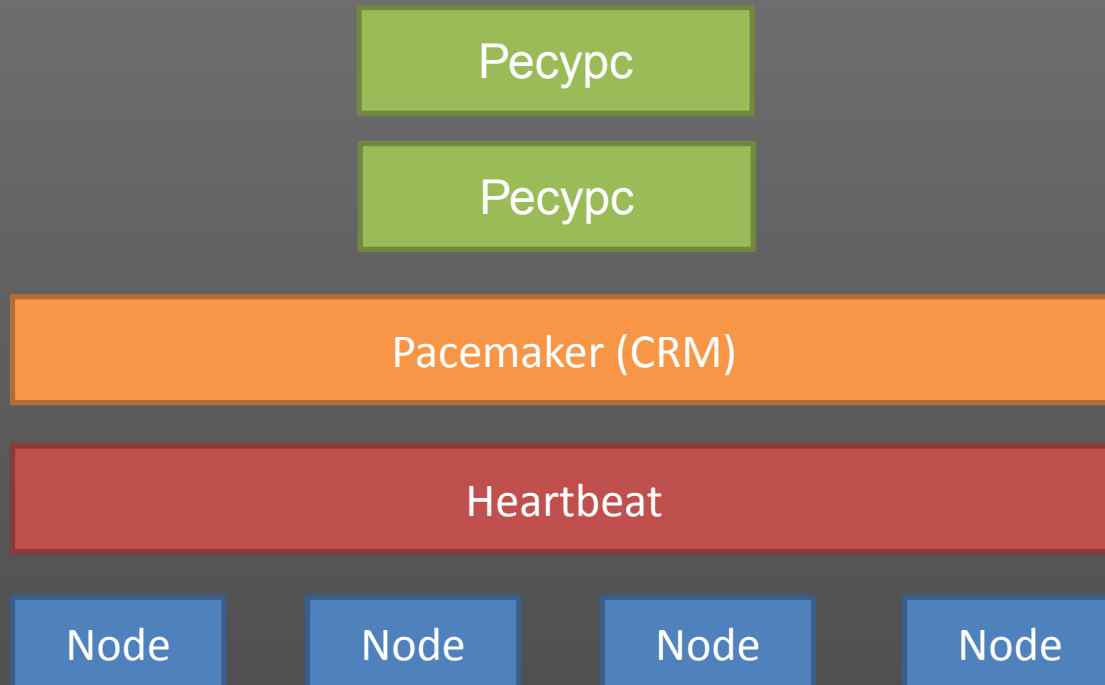
Архитектура HA-Cluster



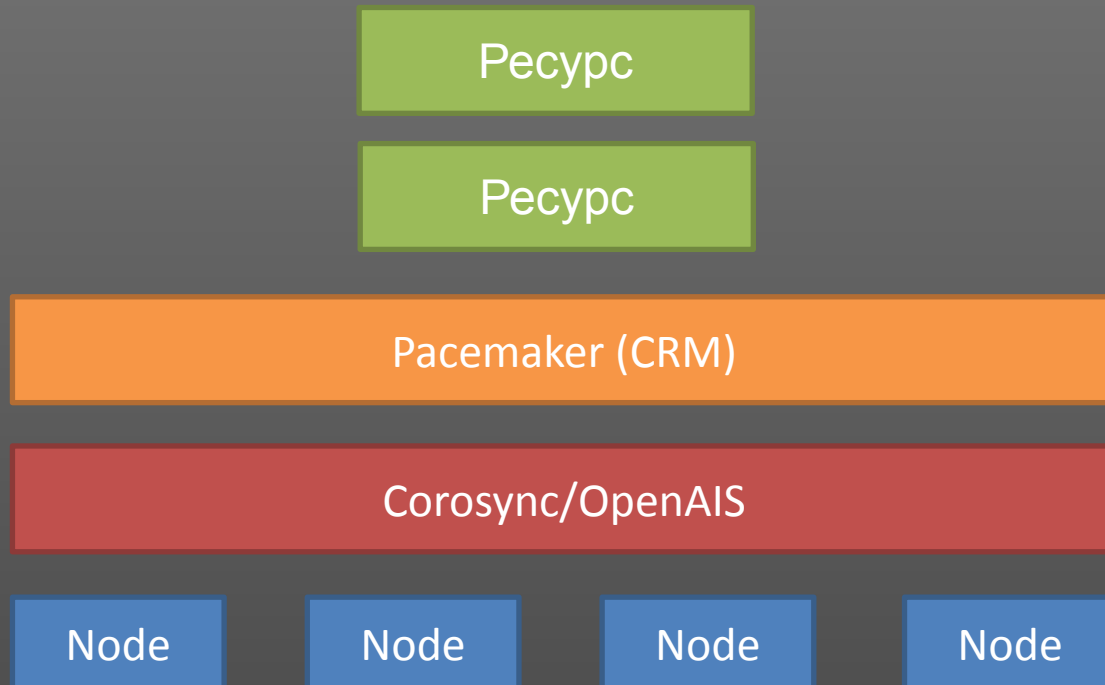
ПО HA-Cluster. Heartbeat



ПО HA-Cluster. Heartbeat + Pacemaker



ПО HA-Cluster. Corosync + Pacemaker



ПО HA-Cluster. Corosync + RGManager



Pecypc

Pecypc

RGManager

Corosync/OpenAIS

CMAN

Node

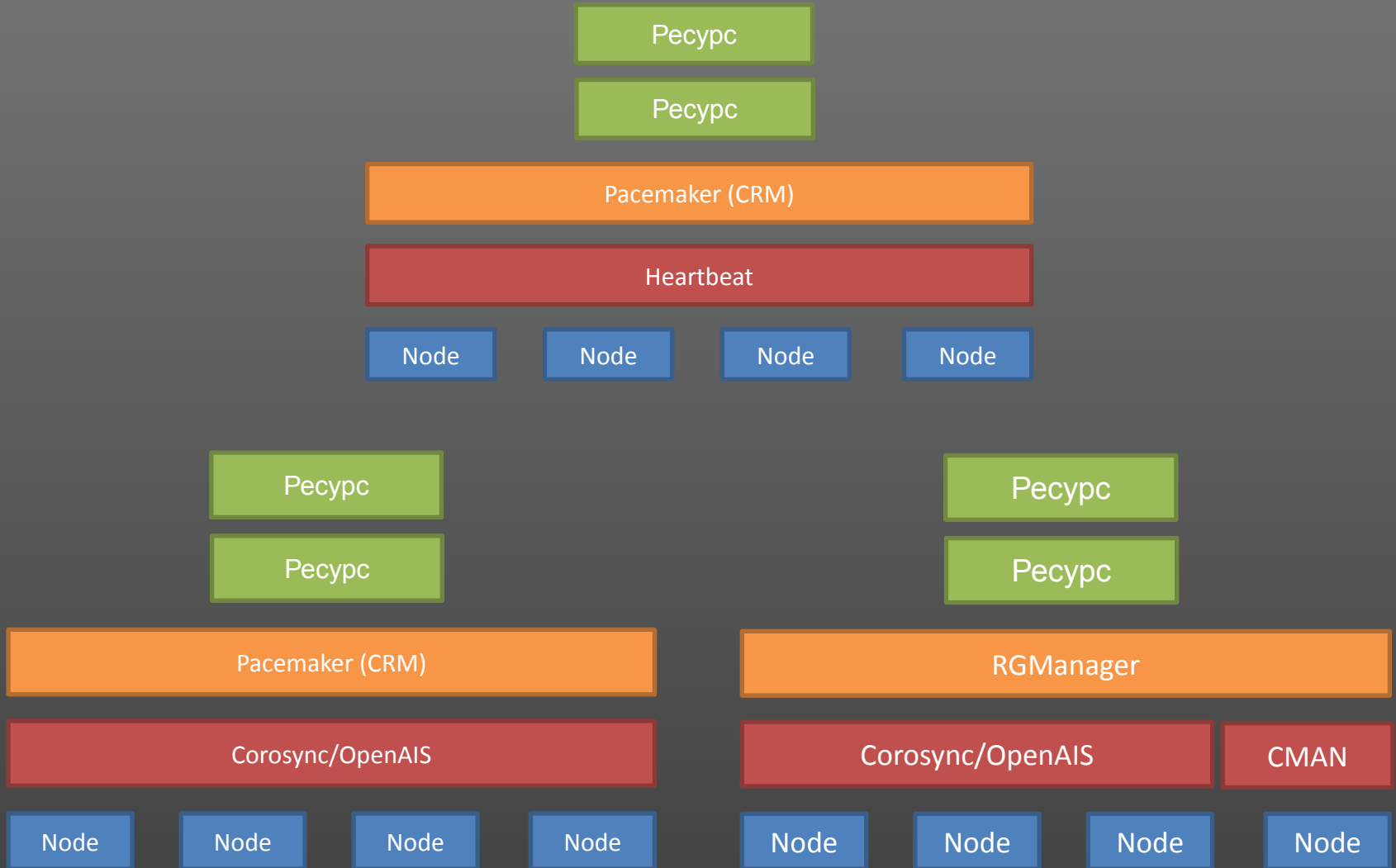
Node

Node

Node

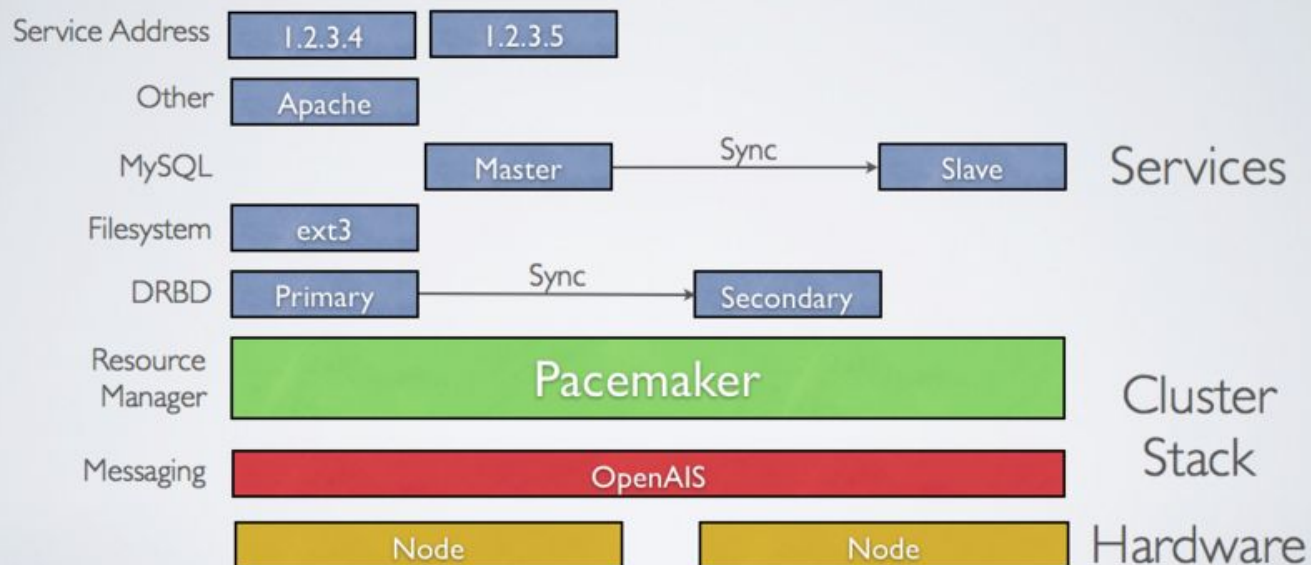
(RedHat Cluster Stack)

ПО HA-Cluster.

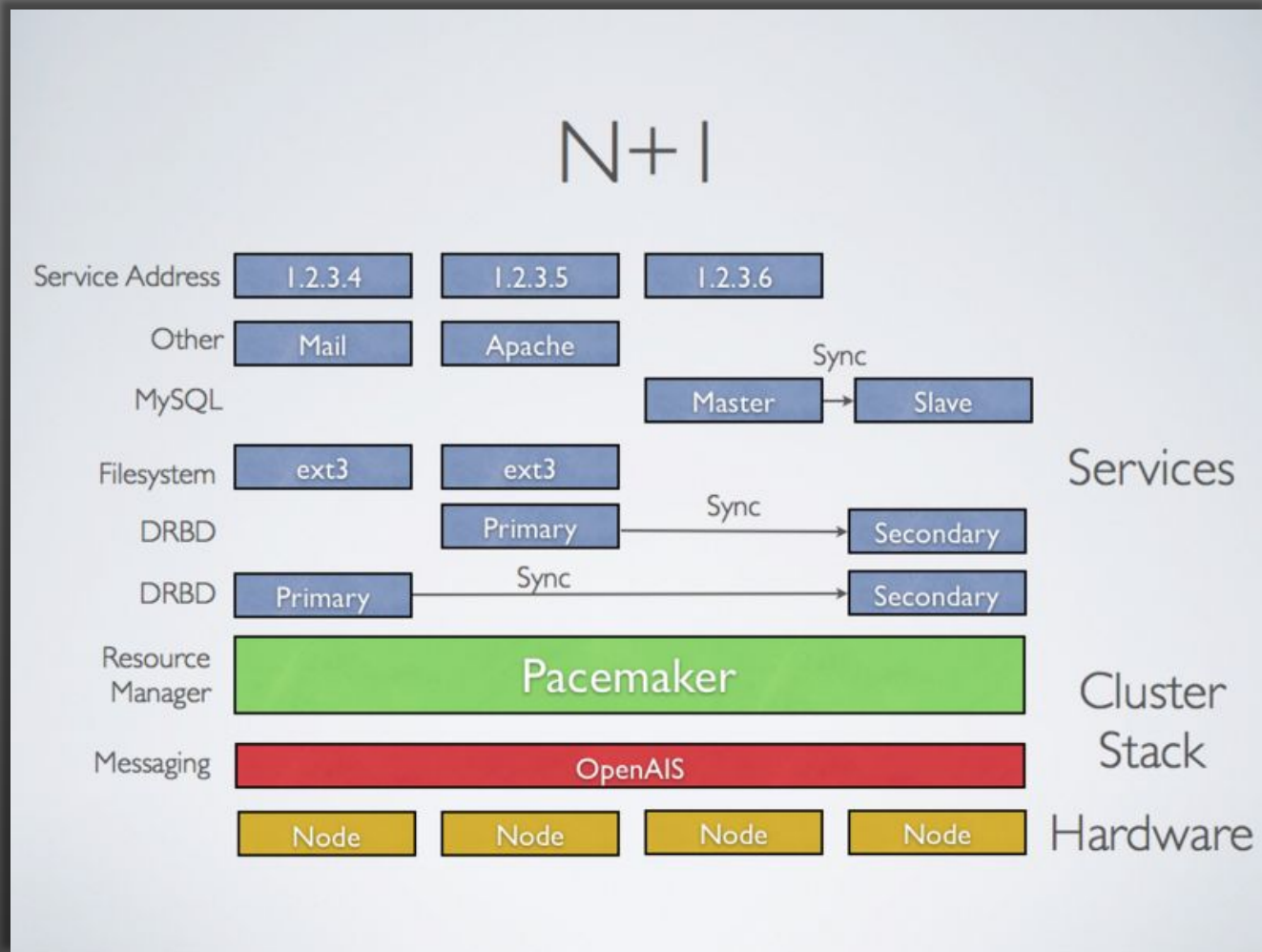


Что умеет Pacemaker

ACTIVE/PASSIVE

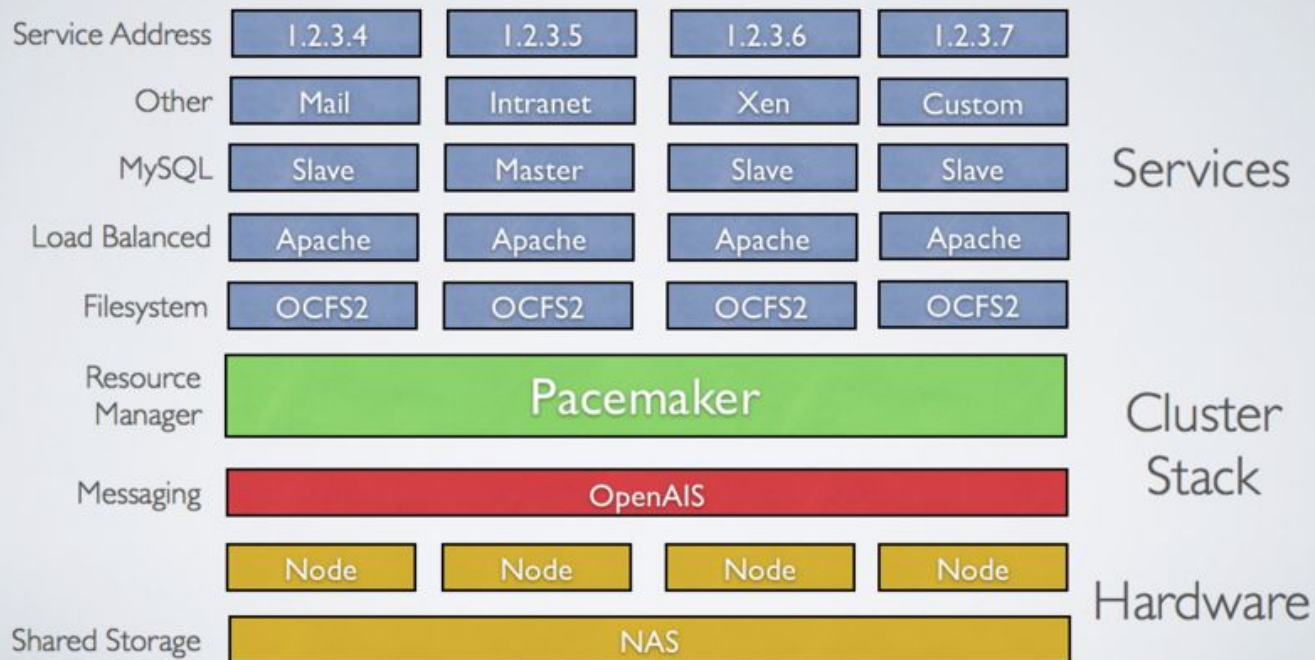


Что умеет Pacemaker

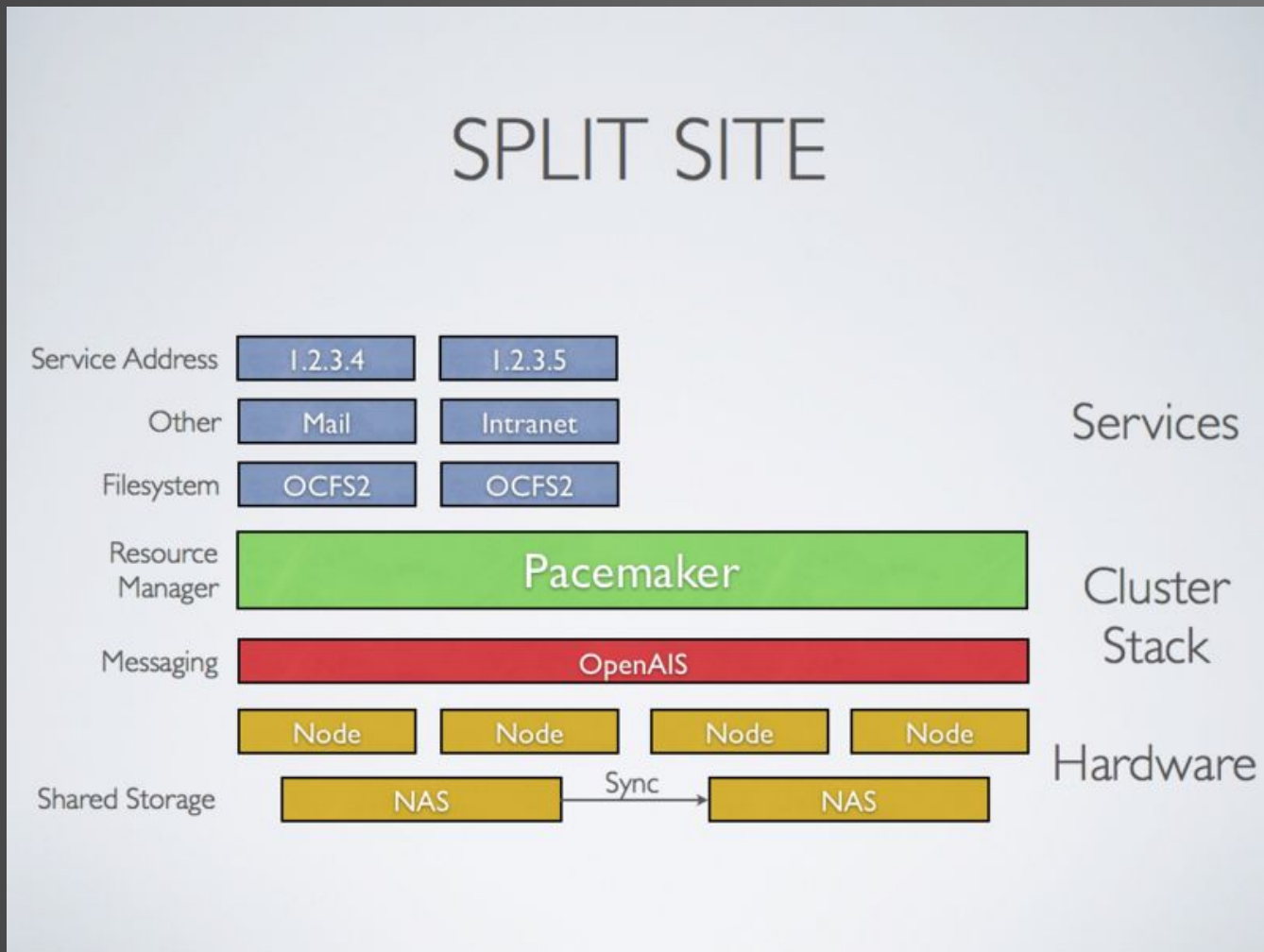


Что умеет Pacemaker

N-TO-N



Что умеет Pacemaker



Задание для получения 5 на экзамене

- Сделать кластер с балансировкой нагрузки, похожий на тот, который демонстрировался на лекции
- Рекомендуемый кластерный стек: Pacemaker + Corosync или Heartbeat
- Рекомендуемый балансир: HAProxy
- Рекомендуемая ОС - Debian
- Виртуальные машины помогут вам (VmWare, VirtualPC, Xen, etc)
- В конце семестра нужно будет продемонстрировать работу кластера и объяснить шаги его настройки

Полезные ссылки

- Clusterlabs.org
- Linux-ha.com

Что почитать:

- Cluster from scratch
- Pacemaker explained

