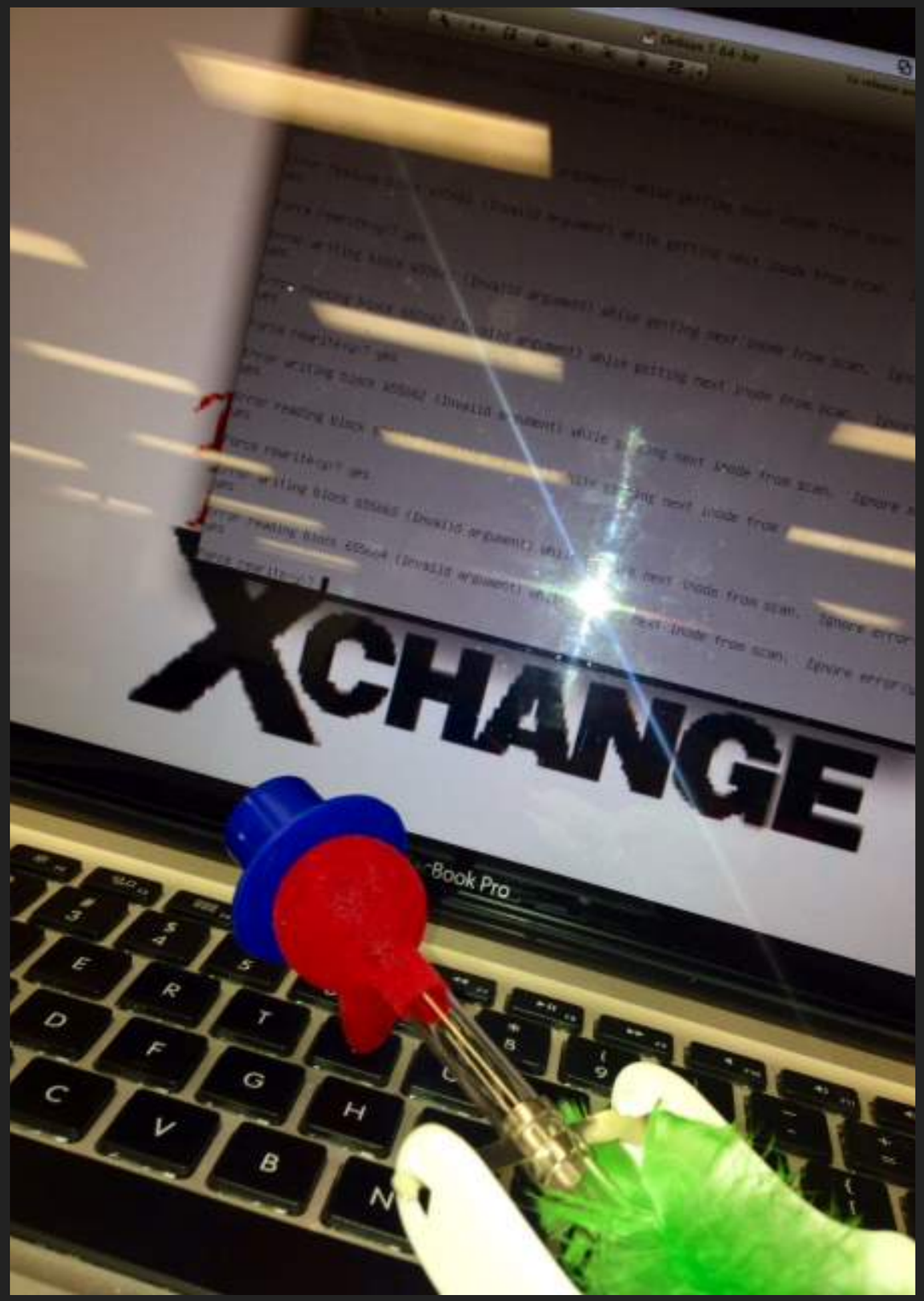
# HOW TO CLUSTER & FAILOVER (ALMOST) ANYTHING

# AN INTRO TO PACEMAKER & COROSYNC

Sam McLeod | https://smcleod.net | @s_mcleod

# WHO AM I?

- Ops Lead @Infoxchange
- 2014 - Researched state of 'enterprise' storage
- 2014 - 'Enterprise' storage sucks
- 2014/2015 - Designed & implemented high performance SSD SANs using standard OSS technologies



**Joel Shea** @realist

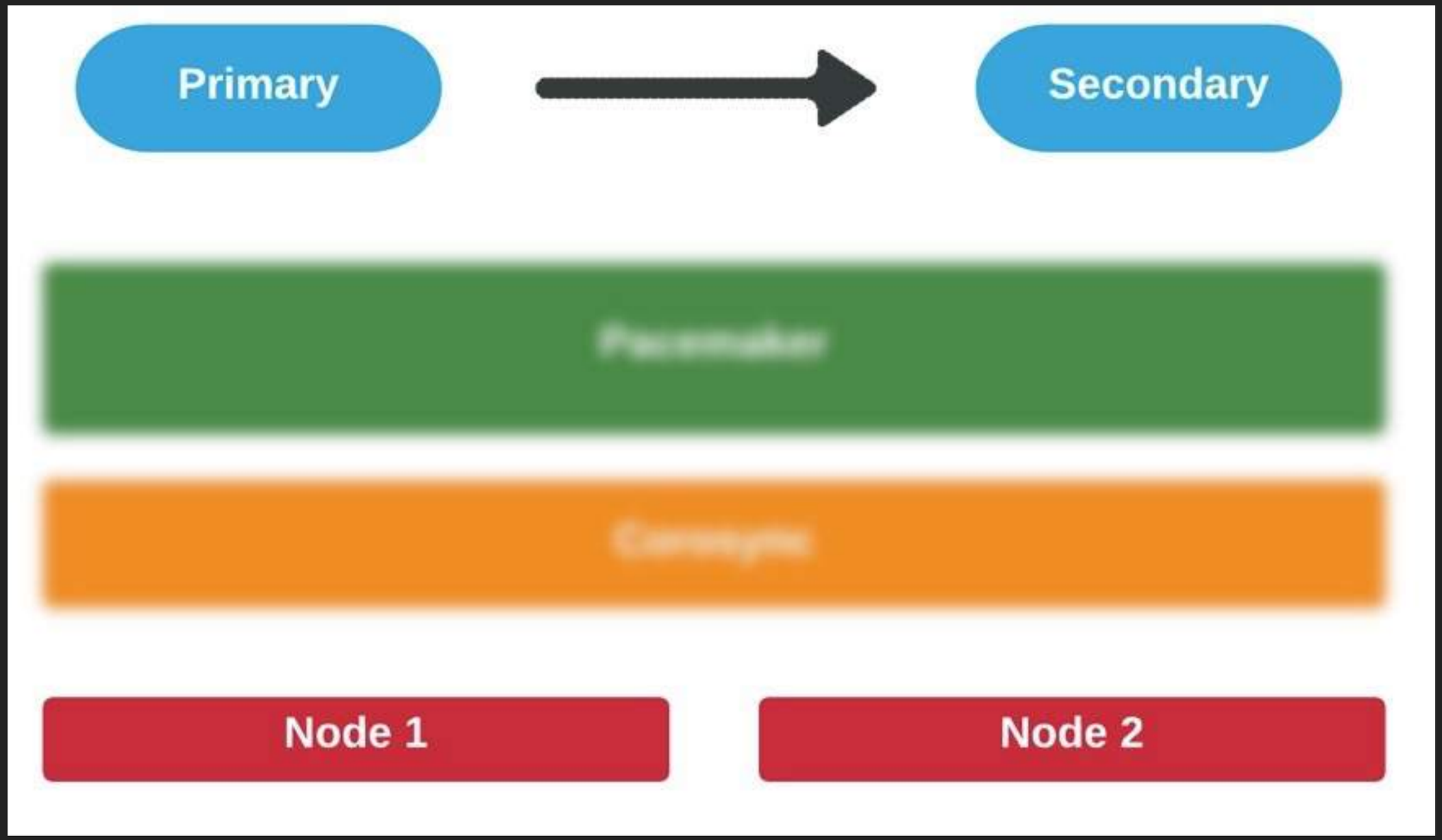
2

. @s_mcleod isn't full of shit when talking redundancy, he even has TP failover at home #devopsdays pic.twitter.com/nA7FRihmeO

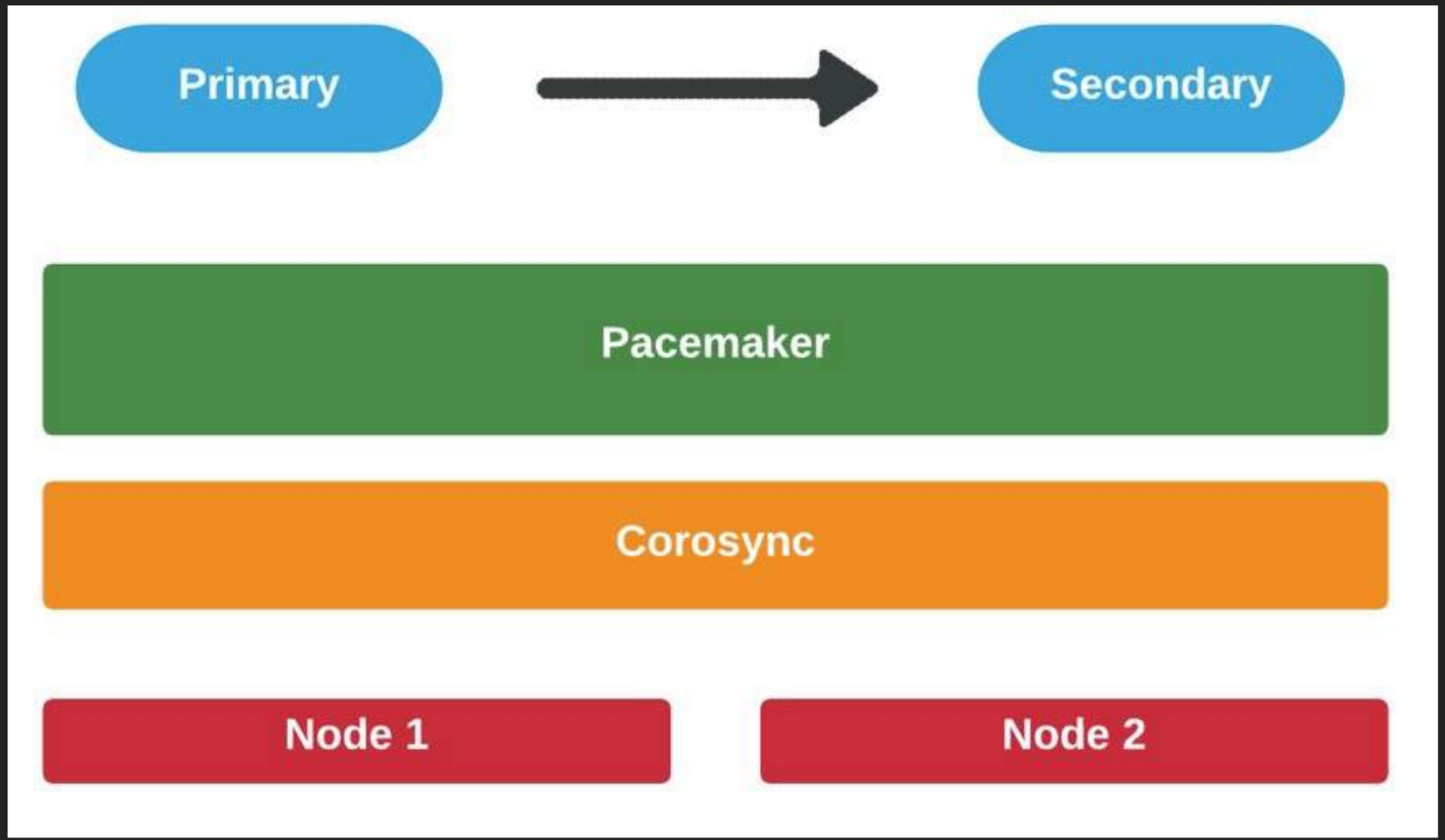
# WHAT CAN YOU CLUSTER WITH COROSYNC+PACEMAKER?

- Floating IPs
- Filesystem / Block Storage
- SystemD / LSB Services
- Database Servers
- VMs (Failover and Load Balancing)
- Network Services

# ACTIVE/STANDBY CLUSTERING



# ACTIVE/STANDBY CLUSTERING



RESOURCE MANAGER

# PACEMAKER

Sam McLeod | https://smcleod.net | @s_mcleod

**Pacemaker is the thing that starts and stops services** (like your database or mail server) and contains logic for ensuring both that they're running, and that they're only running in one location (to avoid data corruption).

**Andrew Beekhof**

http://blog.clusterlabs.org/blog/2010/pacemaker-heartbeat-corosync-wtf/

MESSAGING LAYER

# COROSYNC

Sam McLeod | https://smcleod.net | @s_mcleod

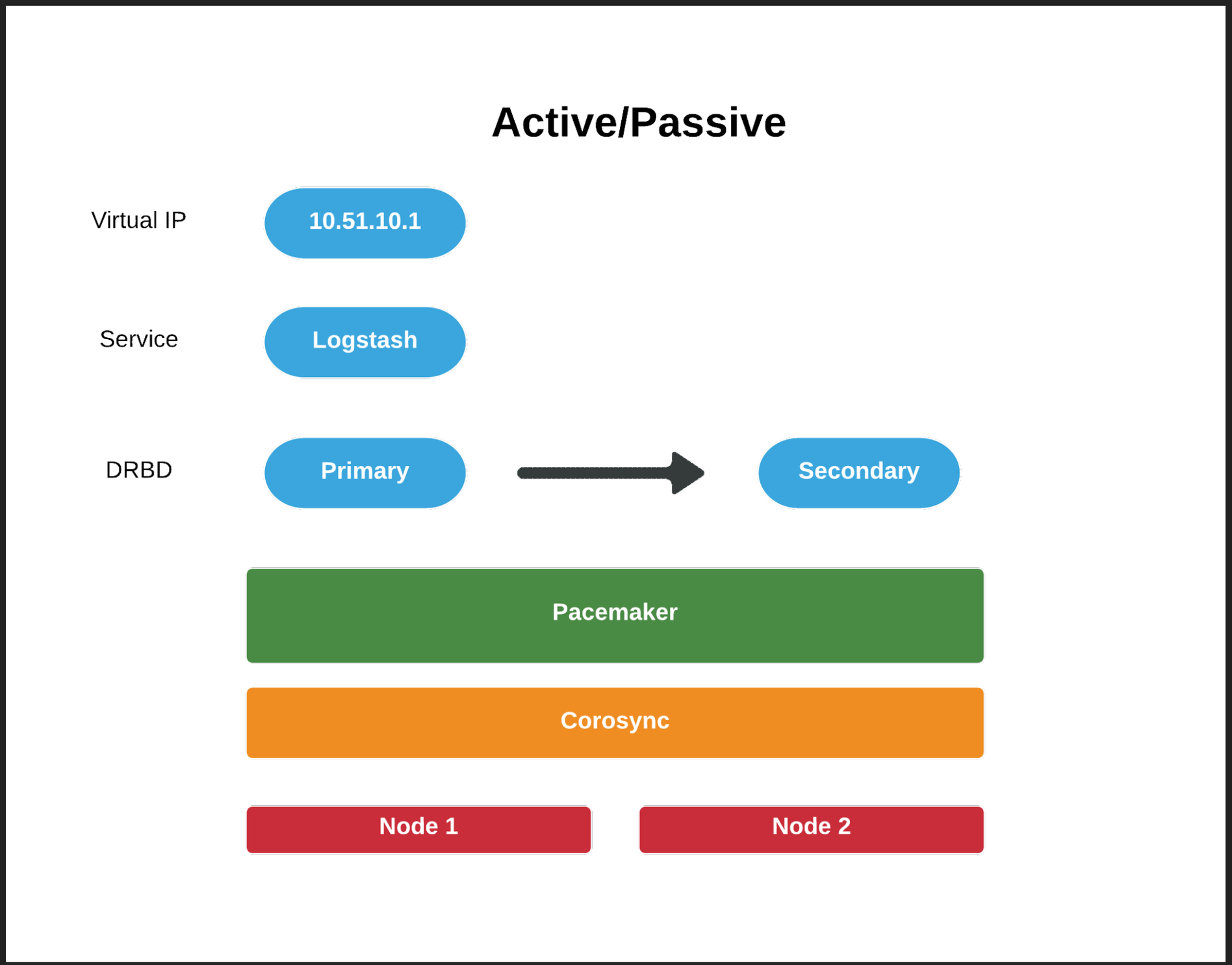
**Think of Corosync as dbus but between nodes.**
Somewhere that any node can throw messages on and know that they'll be received by all its peers.

This bus also ensures that everyone agrees who is (and is not) connected to the bus and tells Pacemaker when that list changes.

**Andrew Beekhof**

http://blog.clusterlabs.org/blog/2010/pacemaker-heartbeat-corosync-wtf/

# FLOATING IP, APPLICATION SERVICE & FILESYSTEM CLUSTER



OPEN CLUSTER FRAMEWORK (OCF)

# RESOURCES

Sam McLeod | https://smcleod.net | @s_mcleod

To avoid teaching Pacemaker about every possible service that people might want to make highly available which we call Resource Agents. **Any series of command-line actions can be easily turned into a resource agent by adding them to an existing template.**

**Andrew Beekhof**

http://blog.clusterlabs.org/blog/2010/pacemaker-heartbeat-corosync-wtf/

# HTTPS://GITHUB.COM/CLUSTERLABS/RESOURCE-AGENTS

```
# pcs resource standards

ocf
lsb
service
systemd
stonith
```

```
# pcs resource agents

CTDB
Delay
Dummy
Filesystem
IPaddr2
LVM
MailTo
```

```
# pcs resource describe IPaddr2

ocf:heartbeat:IPaddr2 - Manages virtual IPv4 and IPv6 addresses
(Linux specific version)
```

# RESOURCE COLOCATION

You can specify that two (or more) resources must be started at the same location (node), in doing so you also define their ordering.

```
# pcs constraint colocation add virtualIP with NFS INFINITY

# pcs constraint colocation add DRBD with RAID INFINITY
```

# RESOURCE ORDERING

Explicit ordering can be set on resources regardless of location

```
# pcs constraint order set virtualIP NFS
```

PACEMAKER + OCF

# RESOURCE MONITORING

Sam McLeod | https://smcleod.net | @s_mcleod

# RESOURCE MONITORING

The frequency that resources are monitored for health as well as how long to wait for them to start / stop and what to do if they fail to perform their action can be added as operational configuration.

```
# pcs resource create myvip IPaddr2 ip=10.51.10.150 cidr_netmask=23 \

op monitor interval=30s timeout=30s \

op start interval=0 timeout=15s on-fail=restart \

op stop interval=0 timeout=15s on-fail=block
```

AVOIDING SPLIT BRAIN

# FENCING & STONITH

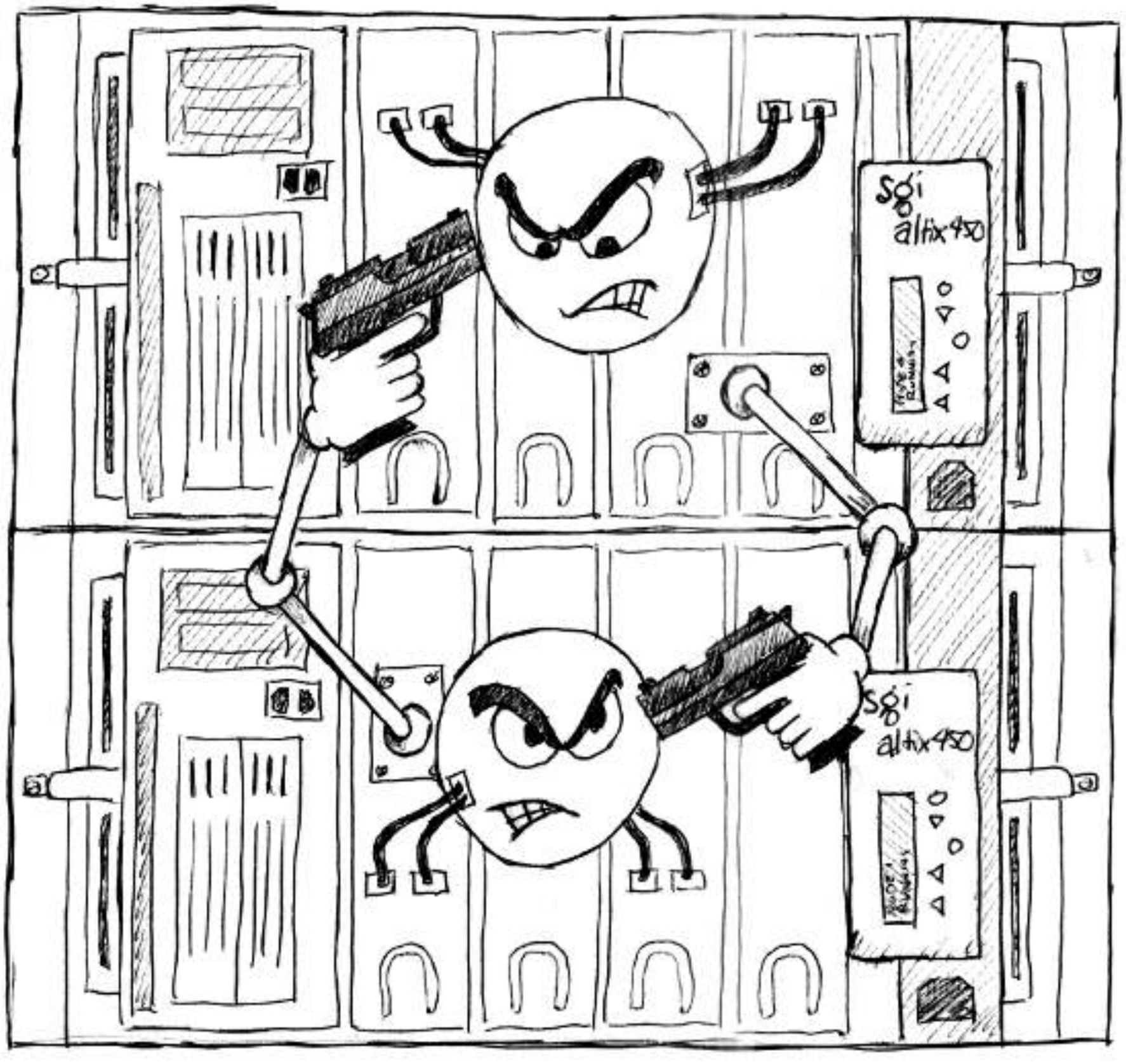
Sam McLeod | https://smcleod.net | @s_mcleod

## STONITH = SHOOT THE OTHER NODE IN THE HEAD

What if a node can't ensure the state of its peer?

The cluster is at risk of split brain.

The peers state is ensured, usually by remote power off or restart.




DON'T ANYBODY MOVE . . .

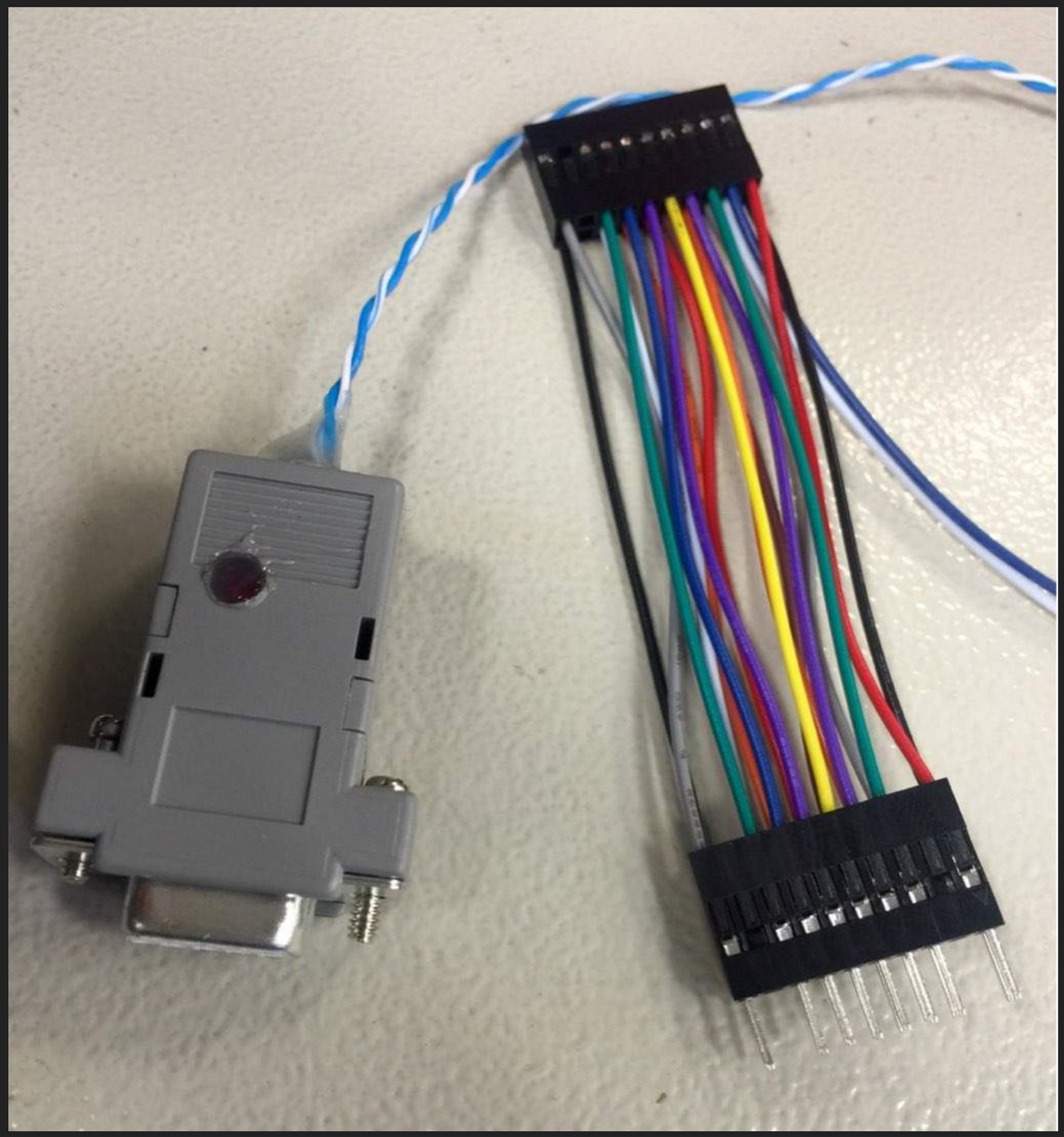
image source: https://ourobengr.com/ha/

# HTTPS://GITHUB.COM/CLUSTERLABS/RESOURCE-AGENTS

```
# pcs stonith list

fence_apc - Fence agent for APC over telnet/ssh
fence_bladecenter - Fence agent for IBM BladeCenter
fence_brocade - Fence agent for HP Brocade over telnet/
ssh
fence_cisco_mds - Fence agent for Cisco MDS
fence_cisco_ucs - Fence agent for Cisco UCS
fence_compute - Fence agent for nova compute nodes
fence_drac5 - Fence agent for Dell DRAC CMC/5
fence_hpblade - Fence agent for HP BladeSystem
fence_idrac - Fence agent for IPMI
fence_ifmib - Fence agent for IF MIB
fence_ilo - Fence agent for HP iLO
fence_ipmilan - Fence agent for IPMI
fence_kdump - Fence agent for use with kdump
fence_rhevm - Fence agent for RHEV-M REST API
fence_scsi - Fence agent for SCSI persistent reservation
fence_virt - Fence agent for virtual machines
fence_vmware_soap - Fence agent for VMWare over SOAP API
fence_xvm - Fence agent for virtual machines
```

# STONITH

# STONITH



CLUSTER DEMO

# PRODUCTION FAILOVER

Sam McLeod | https://smcleod.net | @s_mcleod

rcd_serial (stonith_pm-san6)
running on: pm-san5

rcd_serial (stonith_pm-san5)
running on: pm-san6

online
pm-san5

pacemaker:ping (ping_gateway)
Clone Set (ping_gateway-clone)
running on:
pm-san5 / ping: 1
pm-san6 / ping: 1

online
pm-san6

linbit:drbd (r0)
Master/Slave Set (drbd_r0)
primary on:
pm-san5
secondary on:
pm-san6

linbit:drbd (r1)
Master/Slave Set (drbd_r1)
primary on:
pm-san6
secondary on:
pm-san5

linbit:drbd (r2)
Master/Slave Set (drbd_r2)
primary on:
pm-san6
secondary on:
pm-san5

rd/col

col/ord

IPaddr2 (ip_r0 / 10.51.40.75)
running on: pm-san5

IPaddr2 (ip_r1 / 10.51.40.85)
running on: pm-san6

IPaddr2 (ip_r2 / 10.51.40.95)
running on: pm-san6

iSCSITarget (iscsi_target_r0)
running on: pm-san5

iSCSITarget (iscsi_target_r1)
running on: pm-san6

iSCSITarget (iscsi_target_r2)
running on: pm-san6

iSCSILogicalUnit (iscsi_lun_r0)
running on: pm-san5

iSCSILogicalUnit (iscsi_lun_r1)
running on: pm-san6

iSCSILogicalUnit (iscsi_lun_r2)
running on: pm-san6

anything (iscsi_conf_r0)
running on: pm-san5

anything (iscsi_conf_r1)
running on: pm-san6

anything (iscsi_conf_r2)
running on: pm-san6

PCIe NVMe SSD
PCIe NVMe SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
MDADM RAID 1
MDADM RAID 1+0
MDADM RAID 1+0
LVM
LVM
DRBD
DRBD
DRBD
2x 10GbE CAT6 Bond
(Active/Passive)
iSCSI Target
(Per LUN IP)
iSCSI Target
(Per LUN IP)
iSCSI Target
(Per LUN IP)

Pacemaker Managed

RAID
DRBD
iSCSI Target / LUNs
Virtual IPs

SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
SATA SSD
PCIe NVMe SSD
PCIe NVMe SSD
MDADM RAID 1+0
MDADM RAID 1+0
MDADM RAID 1
LVM
LVM
DRBD
DRBD
DRBD
iSCSI Target
(Per LUN IP)
iSCSI Target
(Per LUN IP)
iSCSI Target
(Per LUN IP)

10Gb DAC
LACP
MEMBER 1
10Gb DAC
LACP
MEMBER 2
Switch Cluster Node 1
Switch Cluster Node 2
10Gb DAC
LACP
MEMBER 1
10Gb DAC
LACP
MEMBER 2

10Gb DAC
LACP
MEMBER 1
10Gb DAC
LACP
MEMBER 2
Switch Cluster Node 1
Switch Cluster Node 2
10Gb DAC
LACP
MEMBER 1
10Gb DAC
LACP
MEMBER 2

iSCSI
Physical Server
LVM
LVM
LVM
VM
VM
VM
iSCSI
Physical Server
LVM
LVM
LVM
VM
VM
VM

iSCSI
Physical Server
LVM
LVM
LVM
VM
VM
VM
iSCSI
Physical Server
LVM
LVM
LVM
VM
VM
VM

# TWO NODE STORAGE CLUSTER

# Two Node Cluster

Design
Performance
Failover Demo

Pause video as required

Sam McLeod
smcleod.net

CLUSTER CREATION

# COROSYNC + PACEMAKER

Sam McLeod | https://smcleod.net | @s_mcleod

Virtual IP **10.51.10.150**

Service **Logstash**

**Pacemaker**

**Corosync**

**dev-samm-01** **dev-samm-02**

Virtual IP

**10.51.10.150**

Service

**Logstash**

**Pacemaker**

**Corosync**

**dev-sa**

Bang!

**dev-samm-02**

```
yum install
pcs corosync pacemaker fence-agents-all
```

# /ETC/COROSYNC/COROSYNC.CONF

```
totem {
version: 2
secauth: off
cluster_name: dev-samm

  interface {
    ringnumber:  0
    bindnetaddr: 10.51.11.231
    broadcast:   yes
  }

}

quorum {
  provider: corosync_votequorum
  two_node: 1
}
```

```
nodelist {

  node {
    ring0_addr: 10.51.11.231
    nodeid: 1
  }

  node {
    ring0_addr: 10.51.11.232
    nodeid: 2
  }

}
```

# /ETC/COROSYNC/SERVICE.D/PCMK

```
service {
  name: pacemaker
  ver:  1
}
```

# START SERVICES

```
systemctl start corosync

systemctl start pacemaker
```

- Starting Corosync does not start Pacemaker, it just sets up the quorum and messaging interfaces needed by the rest of the stack.
- The Pacemaker daemon must be started after Corosync

# CREATE A RESOURCE FOR A FLOATING IP

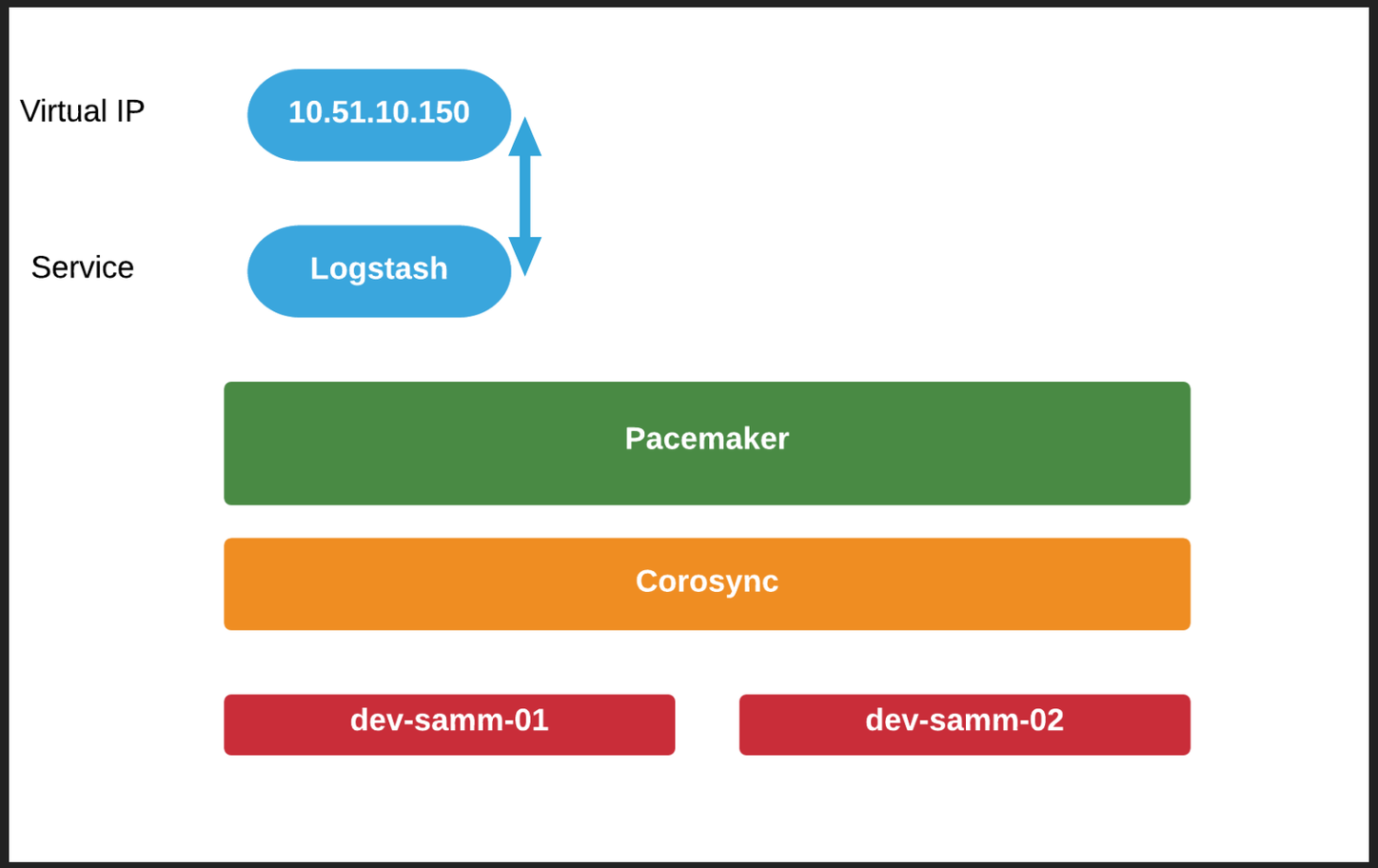
```
# pcs resource create myvip IPaddr2 ip=10.51.10.150
cidr_netmask=23
```

# CREATE A RESOURCE FOR A SYSTEMD SERVICE

```
# pcs resource create logstash systemd:logstash
```

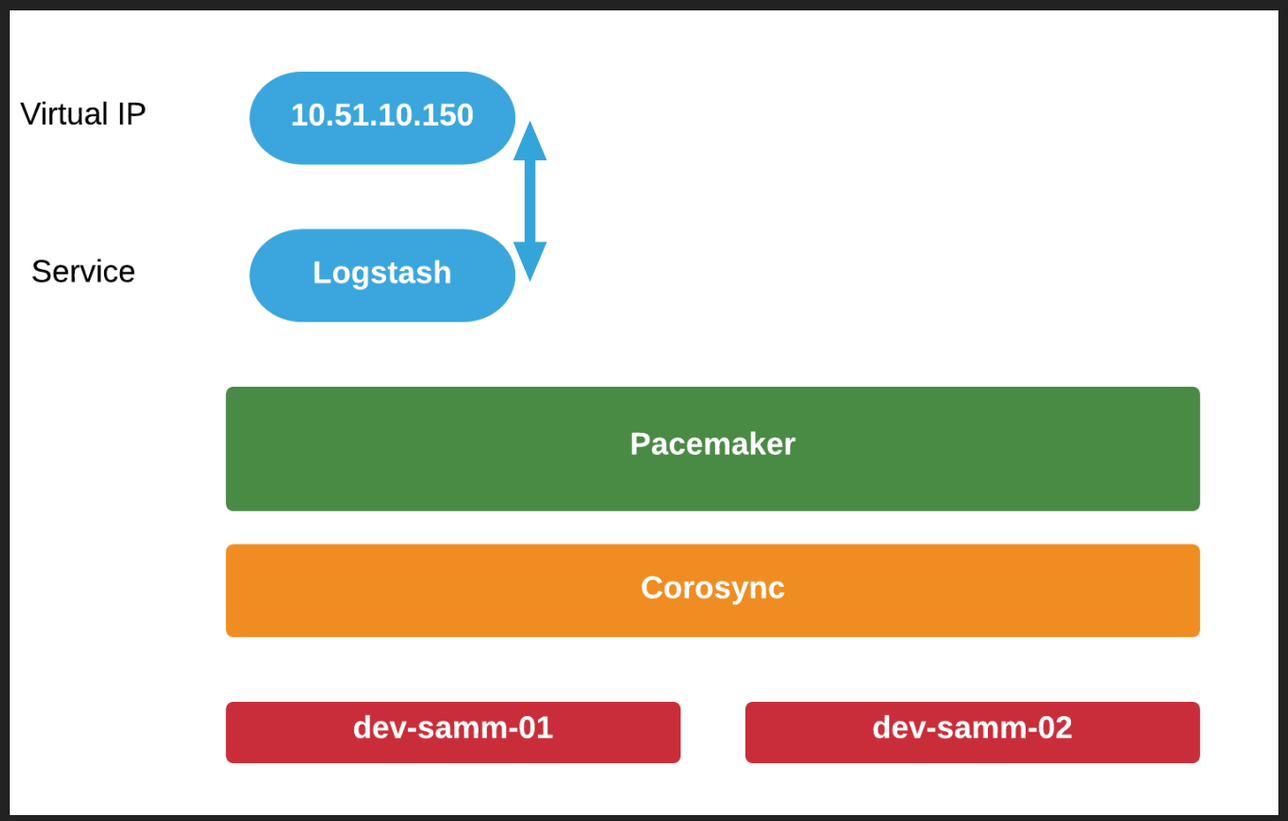
# CREATE A CLUSTER CONSTRAINT THAT COLOCATES THE SERVICE AND VIRTUAL IP

```
# pcs constraint colocation add myvip with logstash
INFINITY
```



# VIEW THE CLUSTER STATUS

```
# pcs status

Online: [ dev-samm-01, dev-samm-02 ]

myvip(ocf::heartbeat:IPaddr2):  Started dev-samm-01
logstash   (systemd:logstash):  Started dev-samm-01
```



Reminder

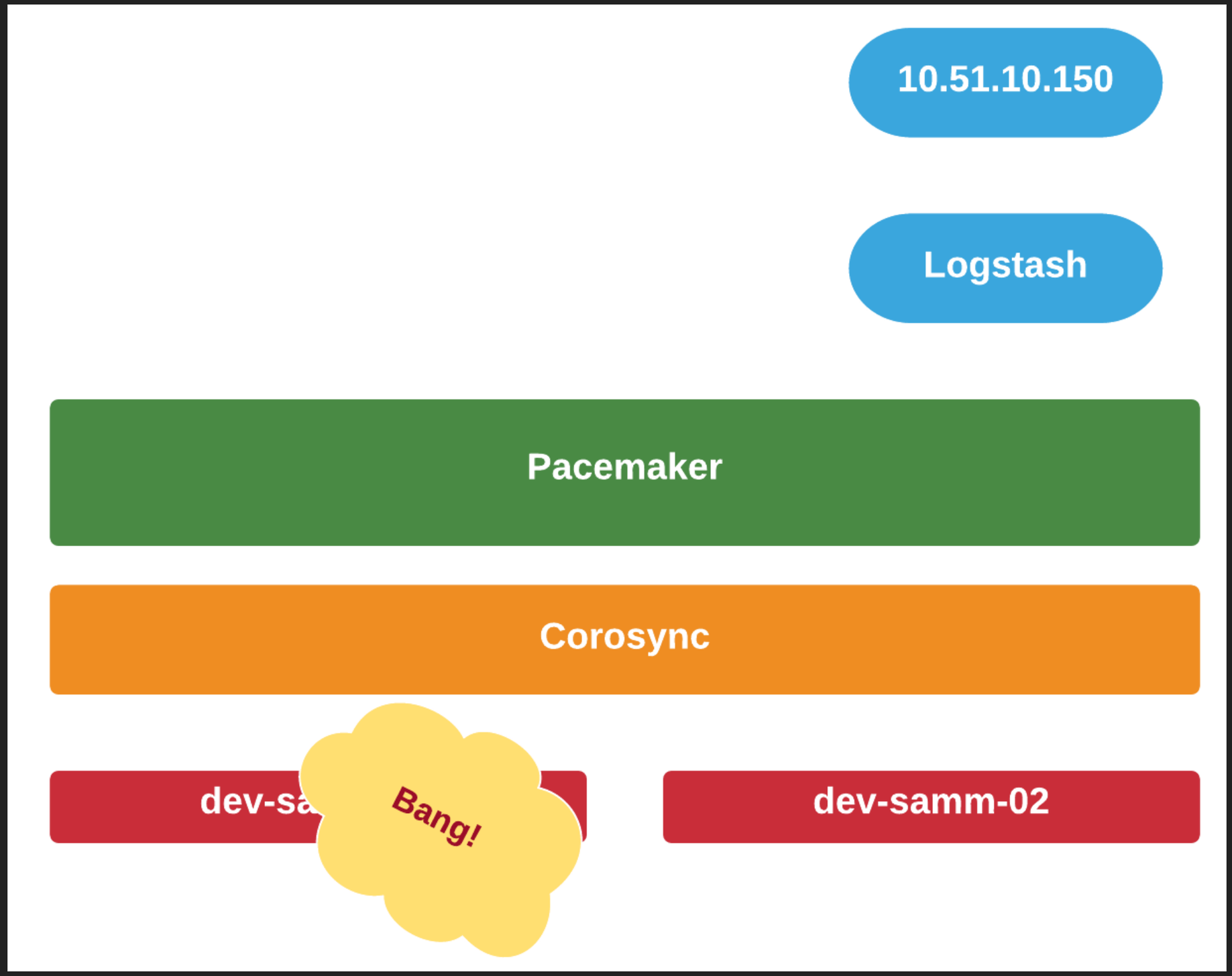
# YOU MUST USE FENCING IN PRODUCTION!

Sam McLeod | https://smcleod.net | @s_mcleod

# TEST BASIC FAILOVER

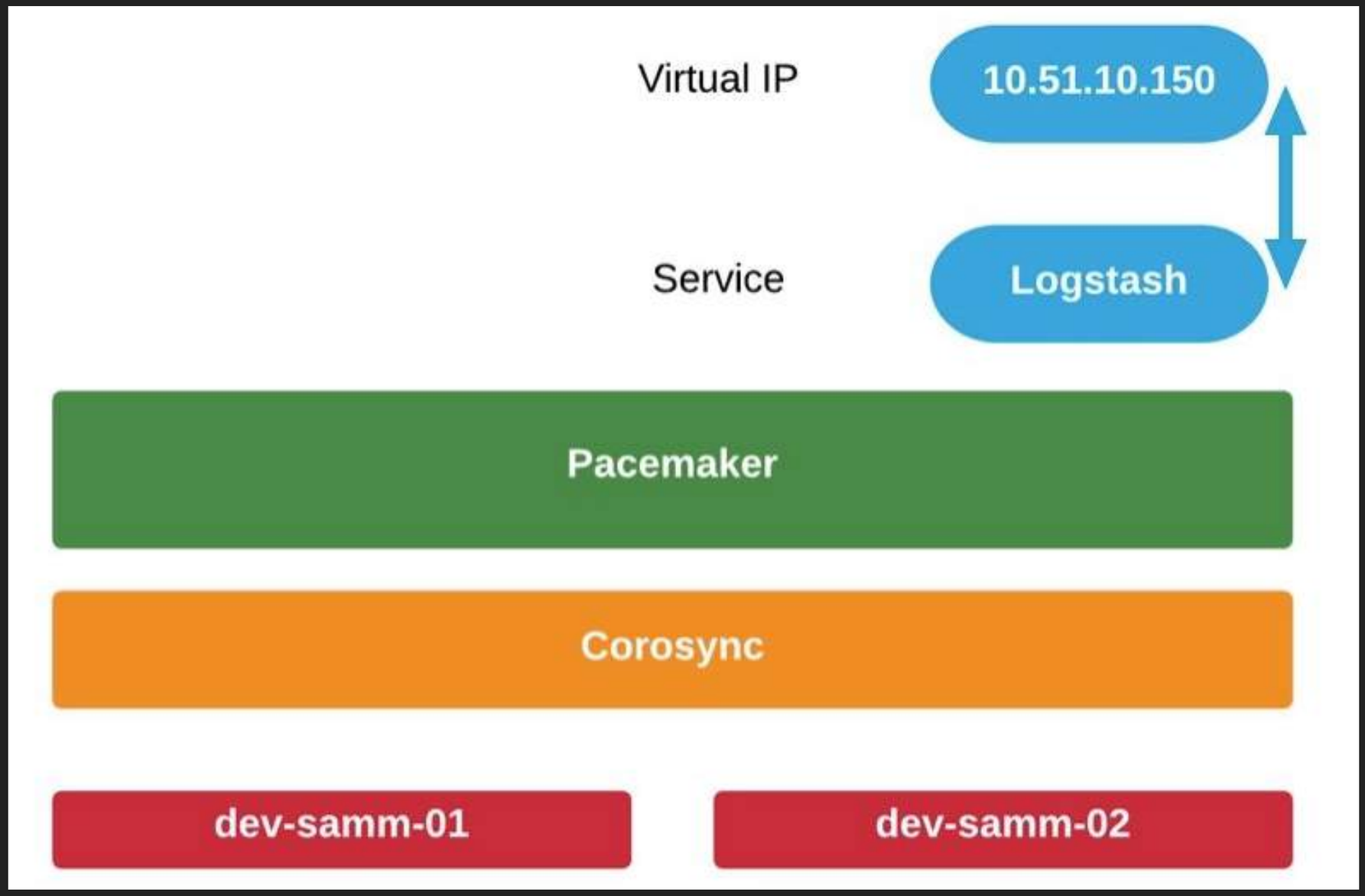
```
# pcs cluster standby dev-samm-01

# pcs status

Node dev-samm-01: standby
Online: [ dev-samm-02 ]
Full list of resources:

 myvip  (ocf::heartbeat:IPaddr2):  Started dev-samm-02

 logstash  (systemd:logstash):  Started dev-samm-02
```

# RESOURCE CREATION - RESOURCE FAILOVER



# TEST BASIC FAILOVER

```
# pcs cluster online dev-samm-01
```



# WHERE DO YOU START?

## CLUSTERS FROM SCRATCH - PACEMAKER 1.1 FOR COROSYNC 2.X AND PCS

# CLUSTERLABS.COM/DOC

Sam McLeod | https://smcleod.net | @s_mcleod

# USEFUL LINKS

- http://clusterlabs.com/doc
- https://alteeve.ca/w/High-Availability_Clustering_in_the_Open_Source_Ecosystem
- https://github.com/ClusterLabs/pacemaker/blob/master/doc/pcs-crmsh-quick-ref.md
- https://github.com/ClusterLabs/resource-agents
- #clusterlabs on FreeNode

Link to these slides

Sam McLeod | https://smcleod.net | @s_mcleod