MySQL High-Availability

with the Percona replication manager (PRM)

Yves Trudeau

April, 2014

MYSQL WORLDWIDE CONFERENCE & EXPO

PERCONA LIVE

About myself : Yves Trudeau

- *Principal architect at Percona since 2009*
- *With MySQL then Sun, 2007 to 2009*
- *Focus on MySQL HA and distributed systems*
- *Database and science background*

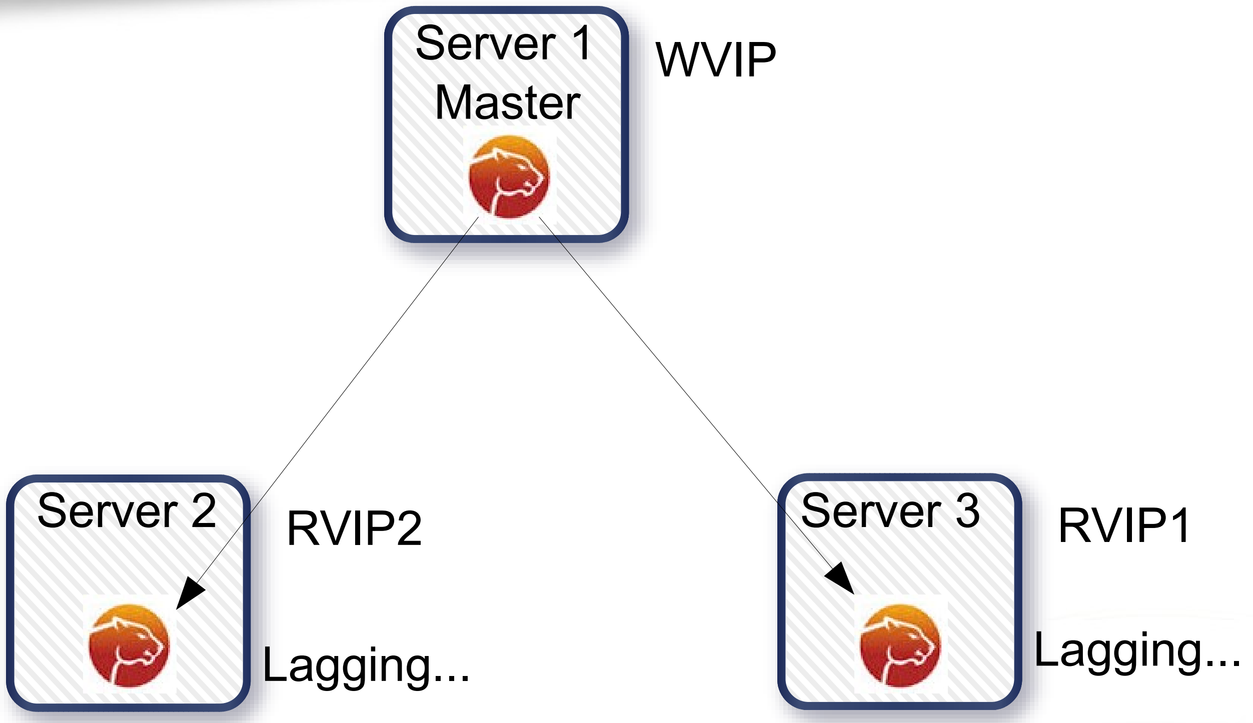
Agenda

- *What is PRM ?*
- *PRM features*
- *Other Percona Pacemaker agents*
- *Example architectures*

What is PRM ?

- *A framework to manage MySQL replication*
- *mysql_prm/Pacemaker/Corosync*
- *A set of Pacemaker rules and the mysql_prm agent*
- *Pacemaker is used for inter-nodes messaging*

What is PRM ? : What it does...



What is PRM ?: Comparison with MHA (1)

MHA:

- *Well know tool, good user base*
- *Nice binlog handling*
- *Complex script in a simple framework*
- *Some limits on HA*

What is PRM ?: Comparison with MHA (2)

PRM:

- *Less known tool, smaller user base*
- *Some binlog handling*
- *Simpler script in a complex framework*
- *Inside pacemaker → full HA*

PRM features : VIPs

- *Manages read and write VIPs*
- *Monitor slaves → RVIP only if OK*
- *Pacemaker custom rules:*
  - *Can avoid cascading read storms*
  - *A reader VIP can have a preferred node*

PRM features : Master handling

- *Score based rules*
- *Master replication info shared to all nodes*
- *Only the master is read/write*
- *Demote → kills existing threads*
- *Monitor sanity → if not OK = demote*

PRM features : Master scores

- *Slaves update their master scores*
  - *based on how far behind they are*
  - *Least far behind = highest score*
  - *Last check done at "pre-promote"*

PRM features : Slave handling

- *After promotion of a master, set replication*
- *When deconfiguring a master, slave is allowed to complete its relay-log*
- *Monitor sanity → RVIP ok*
- *Post-promote script hook for remote slaves*

PRM features : Master hard crashed

- *Slaves are likely not at the same point*
- *Slave last trx hash: af73bd7c5e0c29d2a2c3d7237a7e51e2*
- *New master shares its last 3000 trx:*
- *pos = 3604*

```
2972,2576758f5a131acb408dbb4d7415deaf
3288,af73bd7c5e0c29d2a2c3d7237a7e51e2
3604,e69ddd0831e1fb756e094aba092e01a5
3920,ba6f8412b9dd5bcdf31b09e621037e48
```

PRM features : Custom rules in Pacemaker

- *Manual failover → attribute based*
- *Preferred master(s)*
- *Excluded master*
- *VIP bounded to a slave*
- *Avoid all read VIPs to fall on master*

Other Percona Pacemaker agents: fake_mysql

- *For use when VIPs are not possible*
- *Put on the application servers instead of mysql_prm*
- *Excluded from master role*
- *Upon promotion, redirect traffic to the new master*

Other Percona Pacemaker agents: IPAddr3

- *Improved IPAddr2 for clusterIP*
- *ClusterIP is a special type of VIP using iptables*
- *All nodes have the same VIP but reply based on modulo (remote IP, remote port)*
- *better distribute instances and thus load*

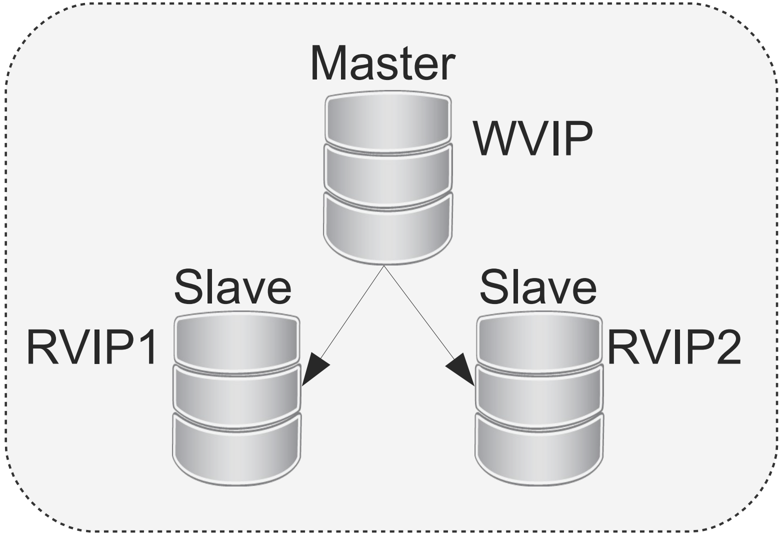
Other Percona Pacemaker agents: mysql_monitor

- *Useful to... monitor mysql*
- *3 modes: pxc, replication, read-only*
- *maintains "readable" and "writable" attributes*
- *Useful with VIPs*
- *check free disk space for PXC 5.5.x*

Other Percona Pacemaker agents: projects

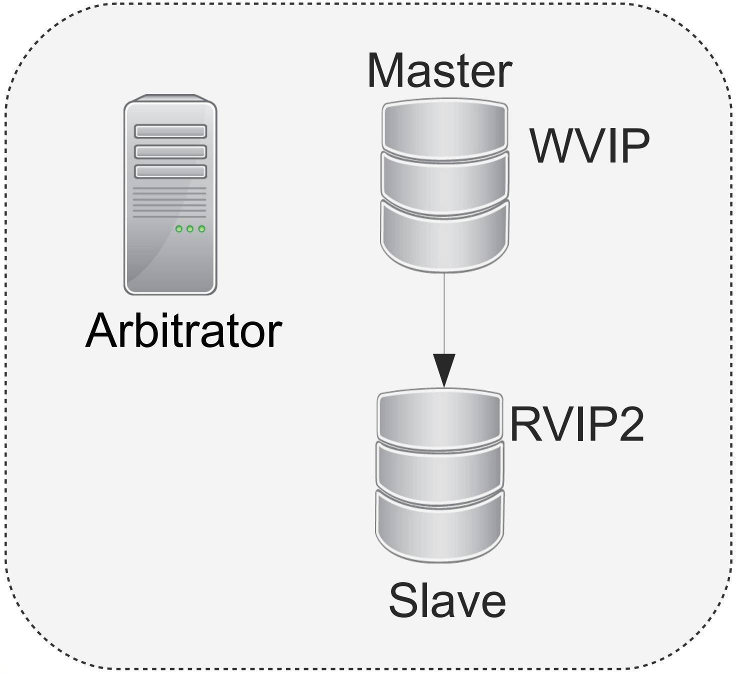
- *IPtablesLB: load balancing using IPtables REDIRECT and DNAT target*
- *IPAddrVPS: VIPs for AWS VPS*
- *PXC: Agent for PXC → bootstrap → Async slaves*
- *Binlog_streamer*

Example configurations: 3 masters



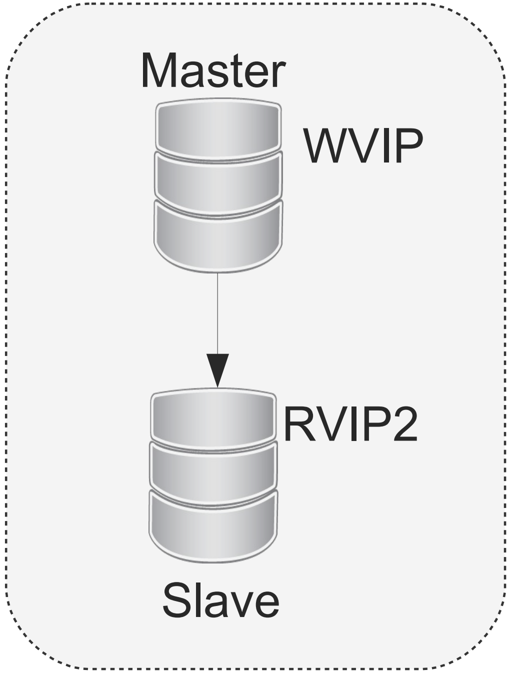
- *3 nodes = Quorum*
- *All nodes are master candidate*
- *Only one is receiving writes*
- *Reads on slave*
- *Preferred/common setup*

Example configurations: 2 masters + arbitrator



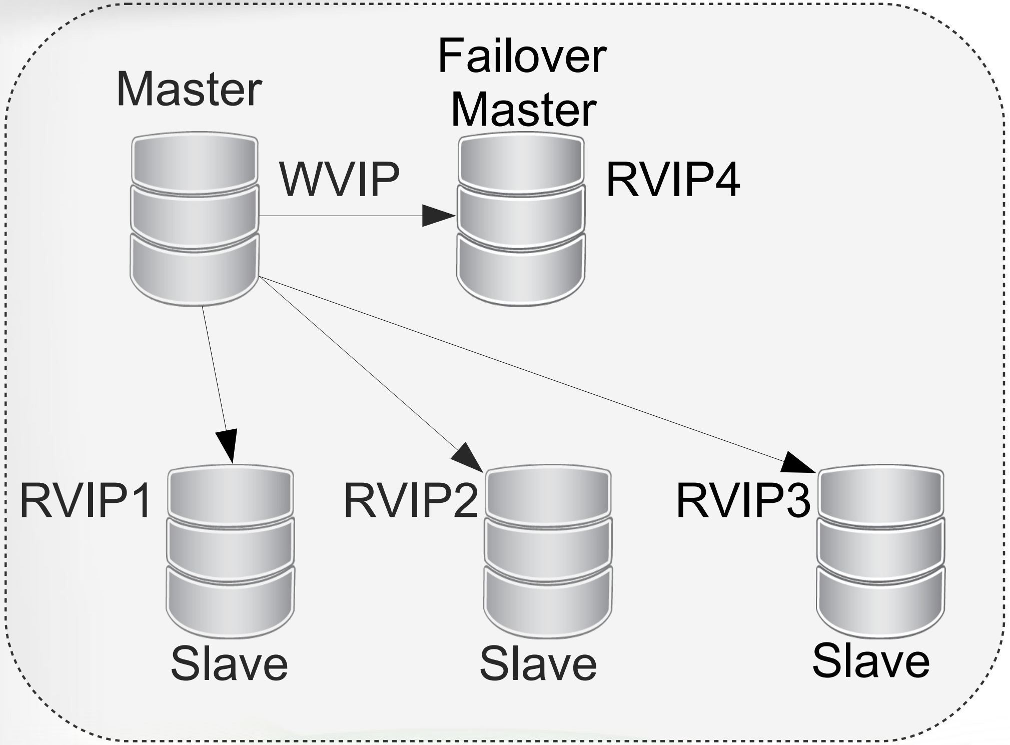
- *3 nodes = Quorum*
- *Arbitrator = Pacemaker in standby*
- *Arbitrator = small box*

Example configurations: 2 masters



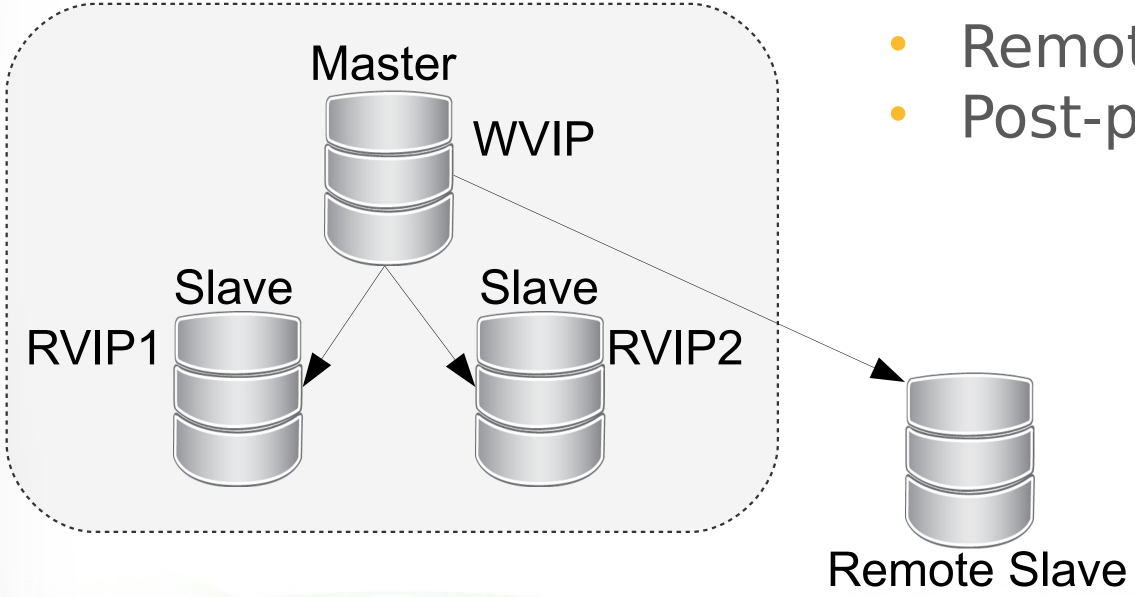
- *2 nodes = no quorum*
- *Not really HA*
- *Could be OK in manual mode*

Example configurations: 2 masters + n slaves



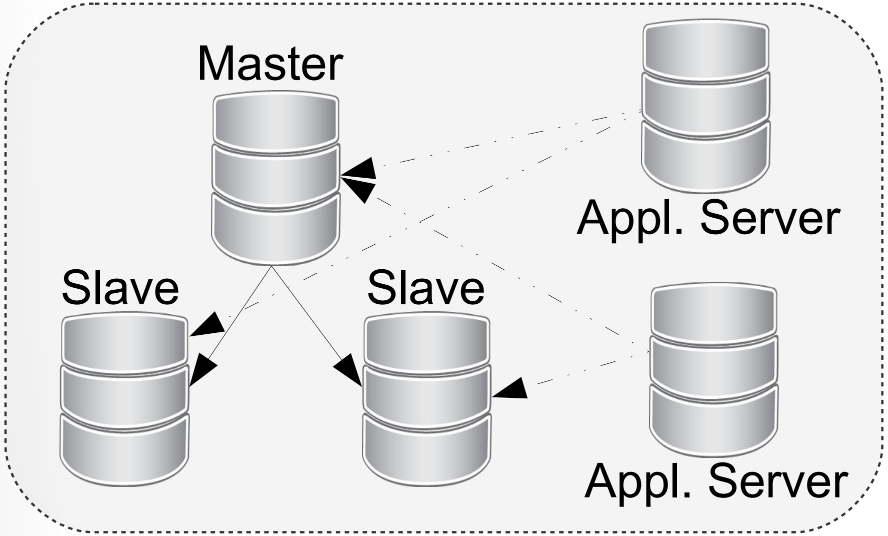
- *Only 2 nodes can be master*
- *Looks like Master-master*
- *Slaves can't be master*
- *Failover master can or not receives read traffic*
- *Odd number of slaves = quorum OK*

Example configurations: External slave(s)



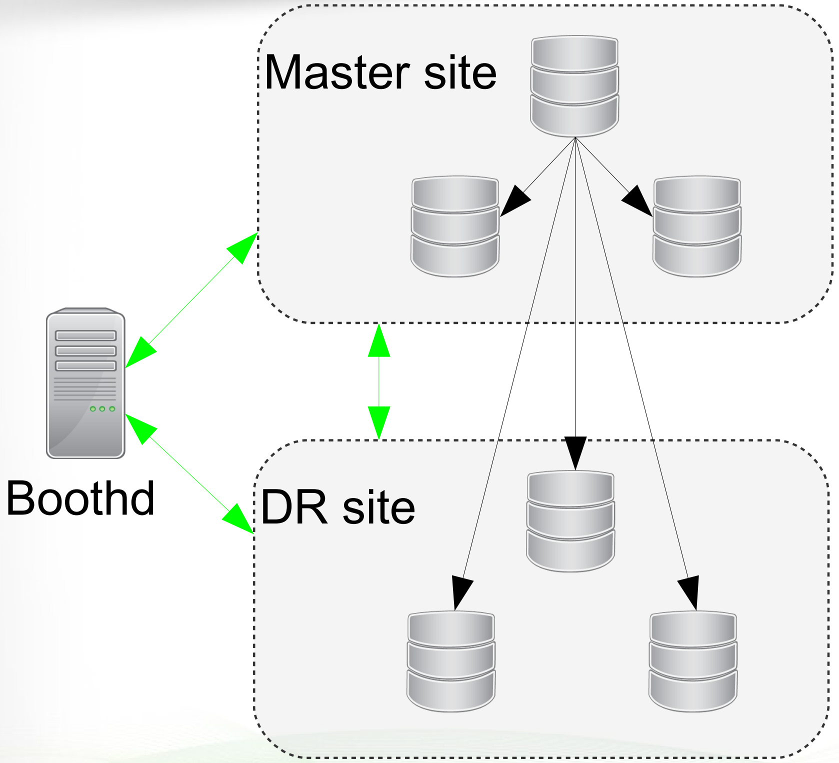
- Remote slave ≠ Pacemaker
- Post-promote script

Example configurations: VIPless environment



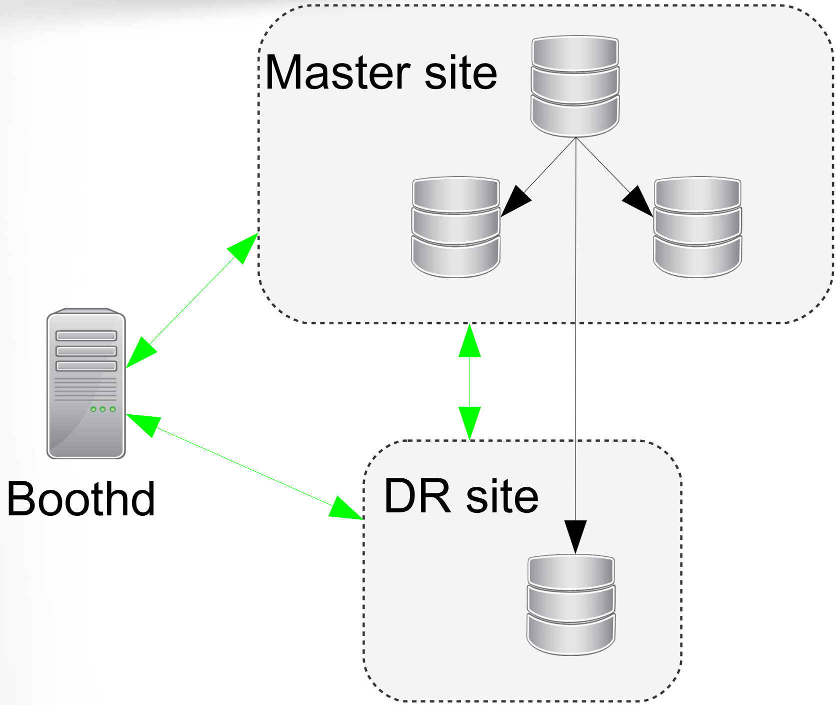
- AWS, Openstack
- Appl. servers = Pacemaker
- fake_mysql agent
- IPtables DNAT, ex:
  - IP: 1.1.1.1 3306 → master
  - IP: 1.1.1.1 3307 → slave

Example configurations: Geo-DR



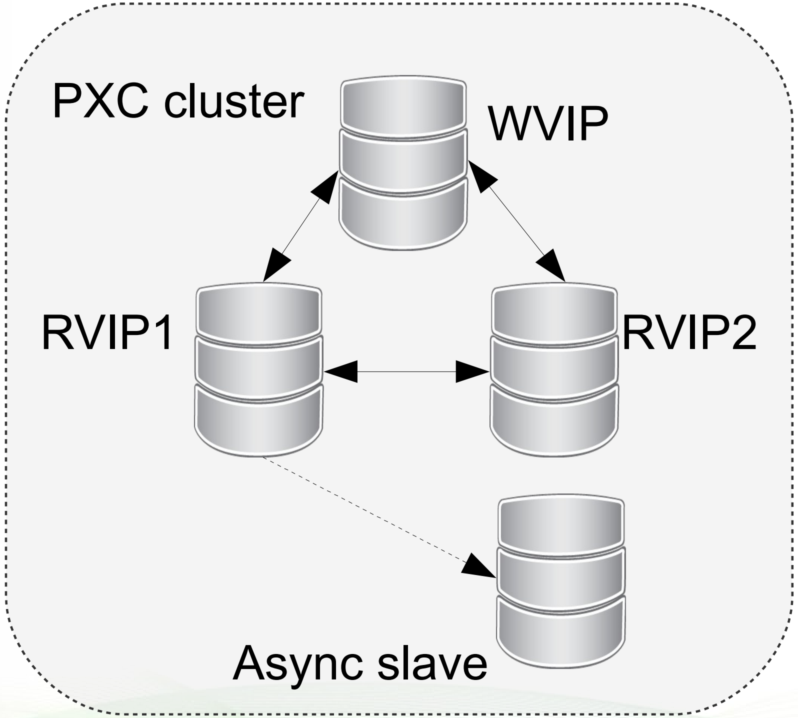
- *2 complete pacemaker clusters*
- *Quorum on each side*
- *Priority to local failover*
- *Need a boothd process elsewhere*
- *Boothd very lightweight*
- *Failover may require DNS change*

Example configurations: Minimal Geo-DR



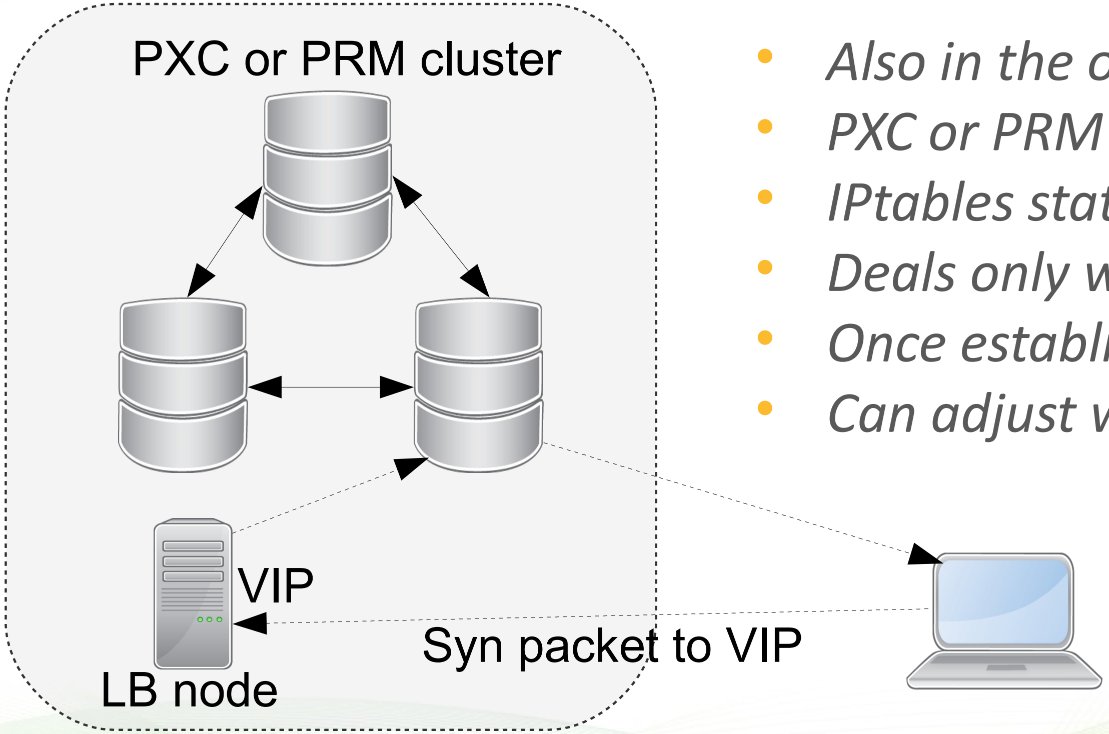
- *Single node pacemaker cluster!!*
- *Need a boothd process elsewhere*

Example configurations: PXC with async slaves



- *In the oven... PXC agent*
- *Bootstrap*
- *Manages VIPs*
- *Async slaves are managed*

Example configurations: IPtables LB



- *Also in the oven...*
- *PXC or PRM*
- *IPtables statistics module*
- *Deals only with SYN packets*
- *Once established = direct*
- *Can adjust with load*

Pitfalls

- *Not always easy to deal with Pacemaker*
- *Swap and overload → killers*
- *Flaky networks*
- *Version mismatch for Corosync/Pacemaker between nodes*

References:

- *GitHUB https://github.com/percona/percona-pacemaker-agents*
- *Setup documentation https://github.com/percona/percona-pacemaker-agents/blob/master/doc/PRM-setup-guide.rst*
- *Geo-DR documentation https://github.com/percona/percona-pacemaker-agents/blob/master/doc/PRM-geo-DR-guide.rst*

Questions?

Questions?