

2TB of RAM ought to be enough for anybody

PG Day FR 2014





Jean-Louis Bergamo - @jlb666 Paris, France

The Presenter – short bio



- 2000-2006 : Easynet
- 2007-2010 : Skyrock.com
 - European biggest blog platform
- 2010-2011 : Winamax.fr
 - #1 Online poker game
- 2011-2012 : Cadremploi.fr
- 2013-03-01: Infra manager





Schibsted Classified Media

From 1 to 30+ Countries in 7 years





















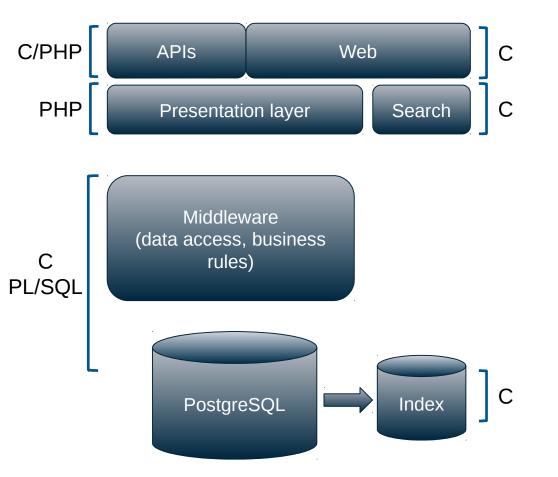








Templated deployment in 30 countries w/ shared technology



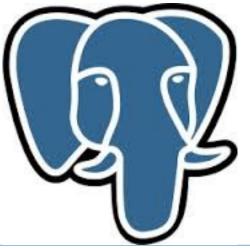
- Technology originally inherited from Blocket.se
- Has since evolved to power all 30 sites
- Focus on performance and ease of local modifications

PostgreSQL in the SCM Platform

100+ servers running PostgreSQL

8TB of data

50+ million classified ads



Schibsted Classified Media & PostgreSQL married...and in love







#1 Classified Web site in France



Project initiated in 2006

Site launch: early 2007



from 1 to 230 people, challenger to #1 in 7 years





Leboncoin.fr part d'une idée simple : la bonne affaire est au coin de la rue ! Pour passer ou chercher des annonces, cliquez sur la région de votre choix et trouvez la bonne affaire parmi **24 612 361 annonces**.

Simple, rapide et efficace !

Alsace Aquitaine Auvergne **Basse-Normandie** Bourgogne **Bretagne** Centre **Champagne-Ardenne** Corse Franche-Comté Haute-Normandie **Ile-de-France** Languedoc-Roussillon Limousin Lorraine **Midi-Pyrénées** Nord-Pas-de-Calais Pays de la Loire **Picardie Poitou-Charentes Provence-Alpes-Côte d'Azur Rhône-Alpes**

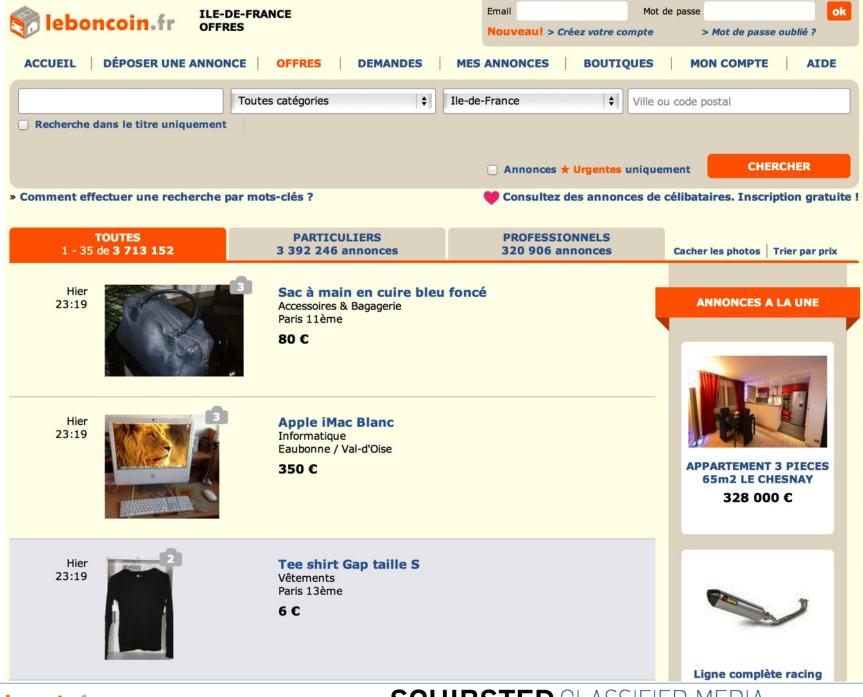
> Guadeloupe Martinique Guyane Réunion

SCHIBSTED CLASSIFIED MEDIA



Déposez gratuitement

vos annonces

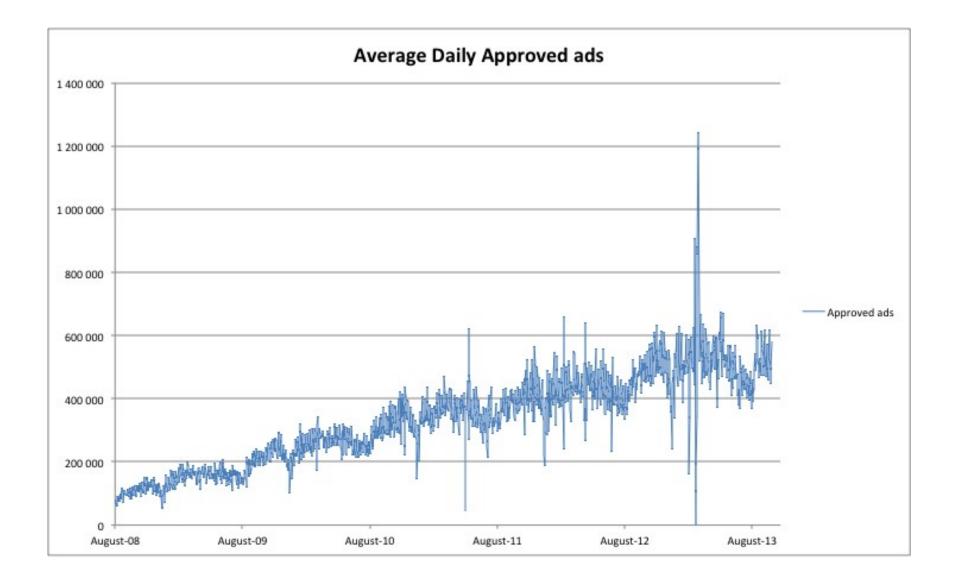








Explosive growth...with a few bumps along the way!





Big Audience

250M page views / day

5M unique visitors / day

18M UV / month

That's 1/3 of French internet population...

600000+ new ads / day

25M live ads

#7 most visited Website in France



Big Ops

300+ servers in 2 DCs

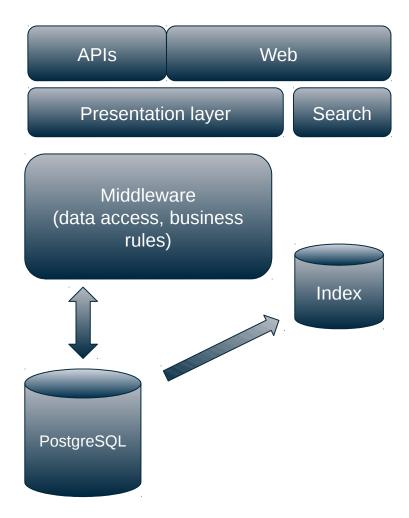
20 servers hosting PG databases

(in production)



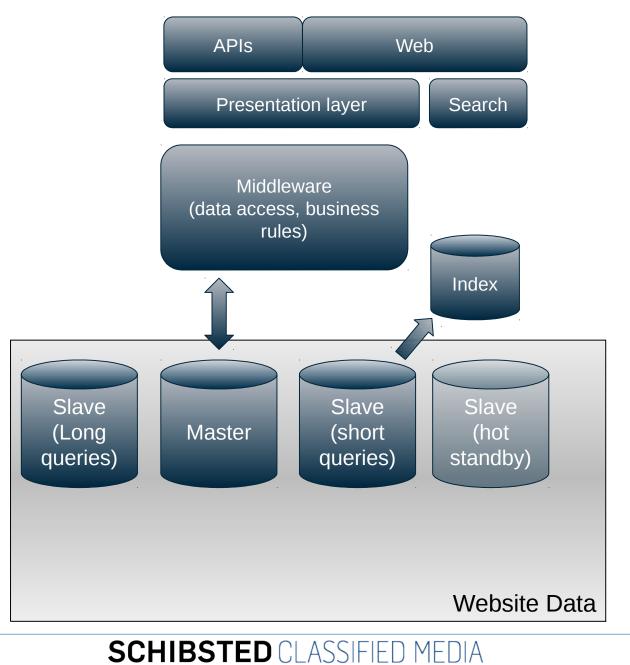


Built on SCM Technology



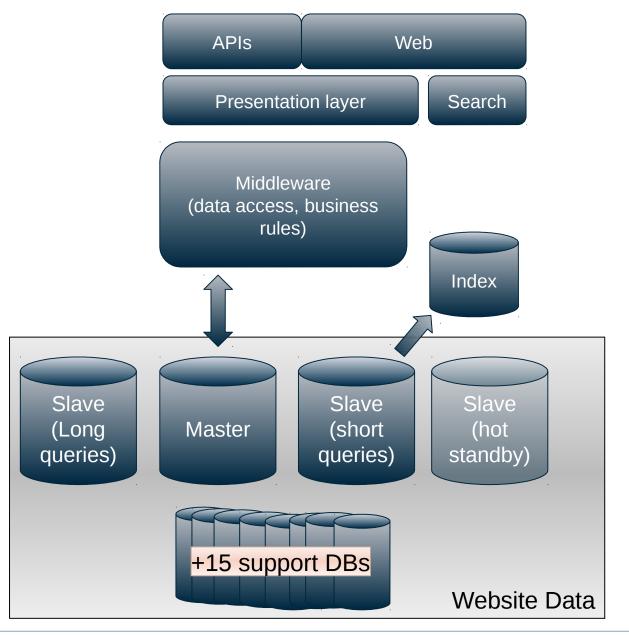


Built on SCM Technology



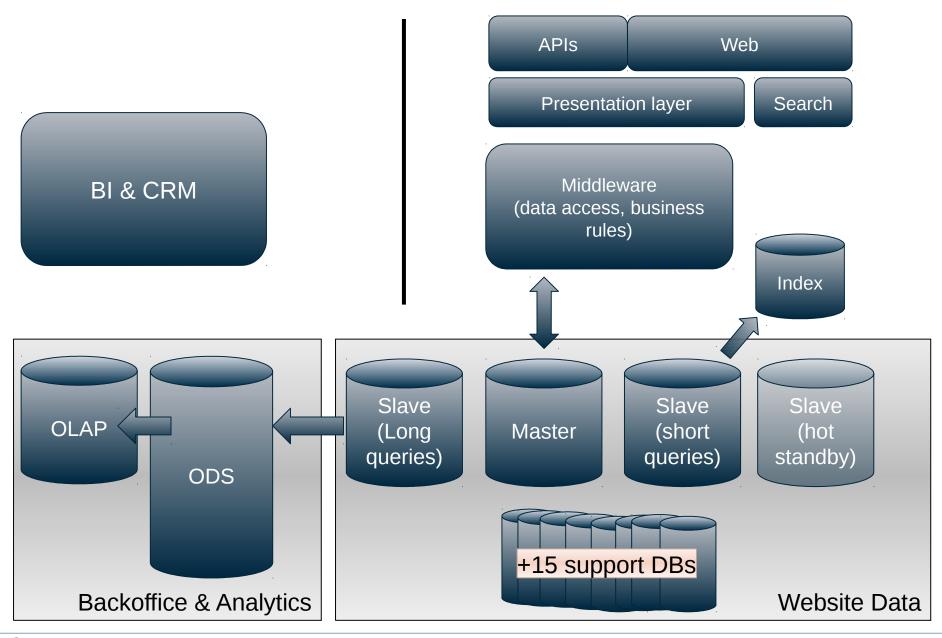


Built on SCM Technology





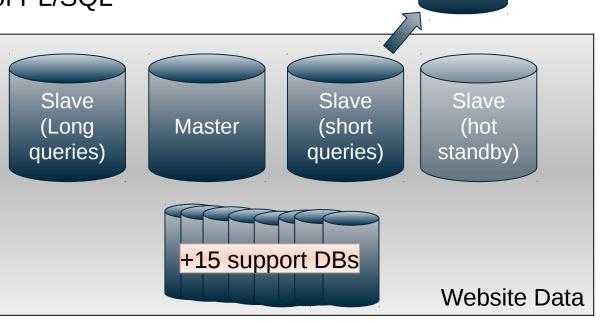
We use PostgreSQL everywhere!





We try to limit writes!

- Index/Search acting as a structured cache
- Master DB workload = 70% writes
 - Slaves used to offload read queries
- Main database = 6TB on disk...
 - +4TB archived away...
- 20K LOCs of PL/SQL



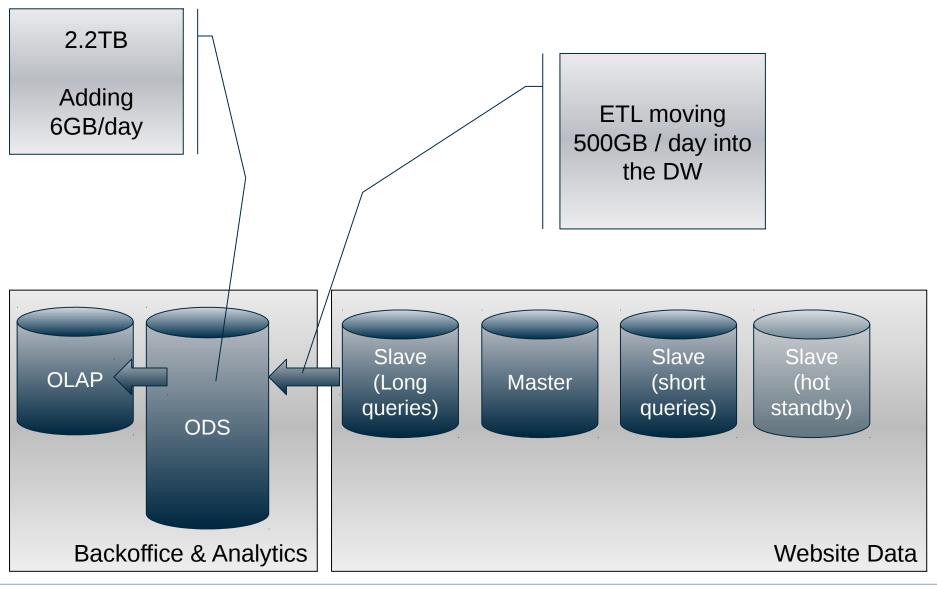
SCHIBSTED CLASSIFIED

Search

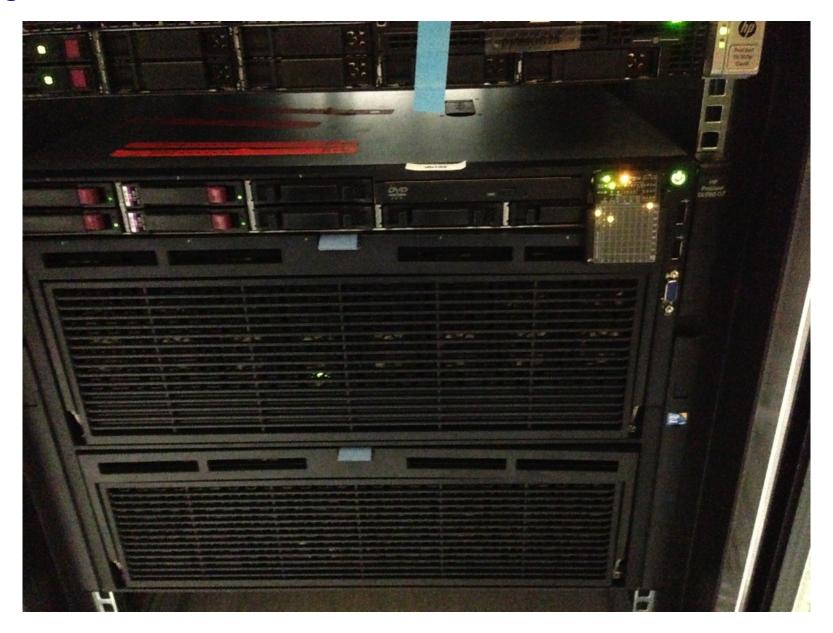
Index



PostgreSQL works beautifully as a DW!



Big Iron: HP DL980, 2TB of RAM, 64-80 cores



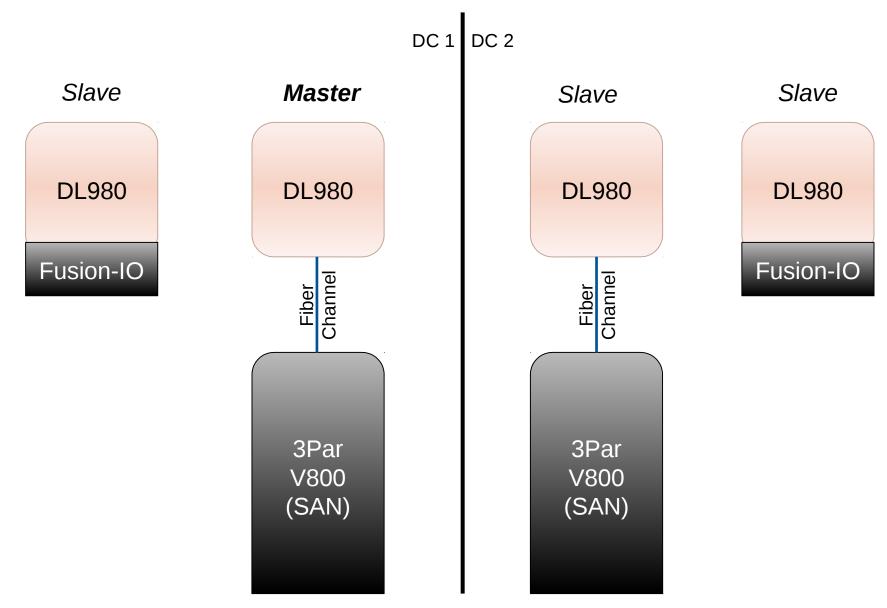


HTOP on the Master

| root@lbc:~ | 🗱 root@vpr | mon01 🗱 | root@db02.leb | 🗱 vi v162.txt | 🗱 root@fwmontc | 🗱 root@oasdb01.l 🗱 | root@db02.leb | ssh ns1.ipsyn.ne | 🗱 root@vpintra04 🗱 | jlb@lbc: ~/git/a | x root@db02.leb | ✗ root@db101.le ✗ |
|--------------------------|---------------------------|---|--|---|--|--|---------------|------------------|--|------------------|-----------------|--|
| 2 | | | 56.0% 59.0% 54.8% 46.7% 47.8% 43.3% 44.0% 57.2% 57.7% 55.0% 51.7% 50.7% 45.3% 44.6% | 18 [] 19 [] 20 [] 21 [] 22 [] 23 [] 23 [] 24 [] 25 [] 26 [] 26 [] 26 [] 27 [] 28 [] 28 [] 29 [] 30 [] 31 [] 32 [] | | 53.5%] 74.4%] 47.8%] 46.5%] 46.3%] 42.5%] 56.9%] 56.8%] 56.8%] 54.7%] 52.0%] 53.7%] 49.8%] 47.8%] 47.8%] 46.8%] (65099/1809673MB] 0/14335MB] | 34 [| | 55.4% 45.5% 37.6% 33.5% 33.5% 62.6% 51.2% 50.2% 33.3% 25.0% 24.7% 15.7% | 52 [| | 56.9%] 54.9%] 52.5%] 48.5%] 47.5%] 48.8%] 44.6%] 50.7%] 57.4%] 53.2%] 49.8%] 51.2%] 40.2%] 45.6%] 37.0%] |
| PID USER 150037 post | PRI NI | VIRT R | ES SHR S <mark>CPU%</mark> 68 4348 R 49.0 | | + Command 8h /usr/bin/python /us | r/hin/ng activity | | | | | | |
| 82716 post | gres 20 0 | | 9M 380M S 21.0 | 0.0 0:10.9 | 91 postgres: trans blo | cketdb 10.4.51.53(602 | | | | | | |
| 83451 post 83335 post | | | | | 63 postgres: trans blo 21 postgres: trans blo | | | | | | | |
| 83444 post 82964 post | gres 20 0 | | 4M 203M S 32.0 | 0.0 0:02.0 | 69 postgres: trans blo 72 postgres: trans blo | cketdb 10.3.51.54(376 | 573) SELECT | | | | | |
| 82964 post 83120 post | | | | | 87 postgres: trans blo | | | | | | | |
| 83094 post | | | | | 47 postgres: trans blo 87 postgres: trans blo | | | | | | | |
| 83389 post 80575 post | | | | | 27 postgres: trans blo | | | | | | | |
| 83449 post | | | | | 60 postgres: trans blo | | | | | | | |
| 83307 post 83043 post | | | | | 62 postgres: trans blo 08 postgres: trans blo | | | | | | | |
| 83320 post | | 25.76 35 | | | 67 postgres: trans blo 23 postgres: trans blo | | | | | | | |
| 83304 post | gres 20 0 | | 3M 607M R 41.0 | 0.0 0:07.0 | 69 postgres: trans blo | cketdb 10.3.51.52(370 | 061) idle | | | | | |
| 76321 post 83357 post | | | | | 02 postgres: trans blo 14 postgres: trans blo | | | | | | | |
| 83436 post | | | | | 08 postgres: trans blo | | | | | | | |
| 83134 post 83443 post | | | | | 22 postgres: trans blo 74 postgres: trans blo | | | | | | | |
| 83167 post | gres 20 0 | | 4M 1693M R 41.0 | 0.1 0:13.3 | 14 postgres: trans blo | cketdb 10.0.2.7(51262 | 2) SELECT | | | | | |
| 83380 post 83392 post | | | | | 37 postgres: trans blo 83 postgres: trans blo | | | | | | | |
| 83069 post | gres 20 0 | | 9M 410M R 35.0 | 0.0 0:13.2 | 22 postgres: trans blo | cketdb 10.3.51.51(41 | 541) SELECT | | | | | |
| 83107 post 83040 post | | | | | 38 postgres: trans blo 80 postgres: trans blo | | | | | | | |
| 83138 post | gres 20 0 | | 6M 378M S 34.0 | 0.0 0:10. | 54 postgres: trans blo | cketdb 10.4.51.51(422 | 288) SELECT | | | | | |
| 83452 post 83286 post | | 25.6G 17 25.7G 28 | | | 40 postgres: trans blo 16 postgres: trans blo | | | | | | | |
| 83361 post | gres 20 0 | | 5M 269M S 10.0 | 0.0 0:05.2 | 24 postgres: trans blo | cketdb 10.3.51.53(59: | 187) idle | | | | | |
| 83390 post 83142 post | | 25.7G 35 25.7G 29 | | | 58 postgres: trans blo 70 postgres: trans blo | | | | | | | |
| 83267 post | gres 20 0 | | 6M 380M S 35.0 | 0.0 0:11. | 58 postgres: trans blo | cketdb 10.4.51.54(532 | 248) SELECT | | | | | |
| 83160 post 81322 post | | | | | 98 postgres: trans blo 69 postgres: trans blo | | | | | | | |
| 83128 post | gres 20 0 | | 3M 326M S 18.0 | 0.0 0:08.2 | 26 postgres: trans blo | cketdb 10.4.51.52(609 | 953) SELECT | | | | | |
| 83110 post 83381 post | | | | | 54 postgres: trans blo 08 postgres: trans blo | | | | | | | |
| 70023 post | <u>gres</u> 20 0 | 26.0G 953 | <u>3M 9145M S 36.0</u> | 0.5 4:56.0 | 04 postgres: postgres | | | | | | | |
| F1Help F2Se | tup <mark>B</mark> Search | 4 <mark>Filter</mark> F5 <mark>T</mark> | ree <mark>F6</mark> SortBy <mark>F7</mark> | Nice - <mark>-8</mark> Nice | +F9Kill F10Quit | | | | | | | |

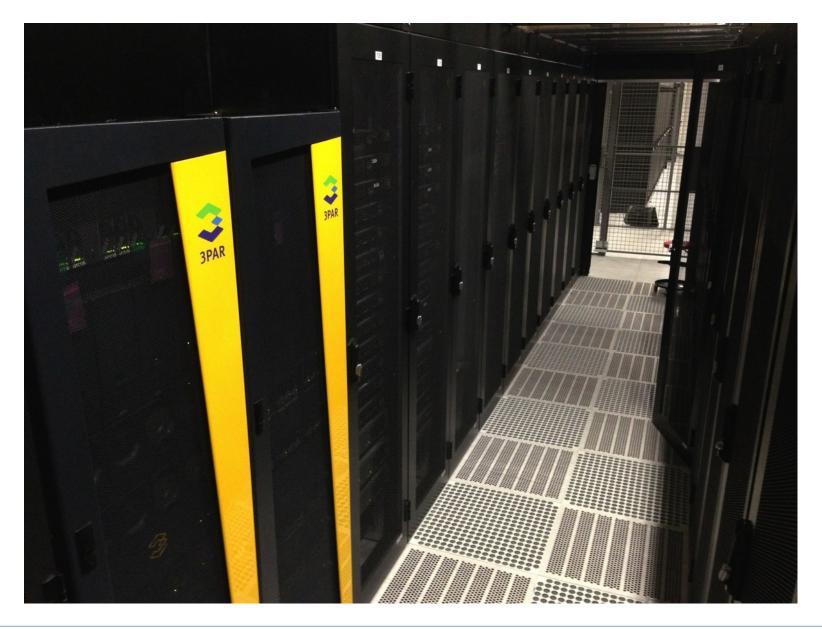


Physical storage for the main PostgreSQL instances



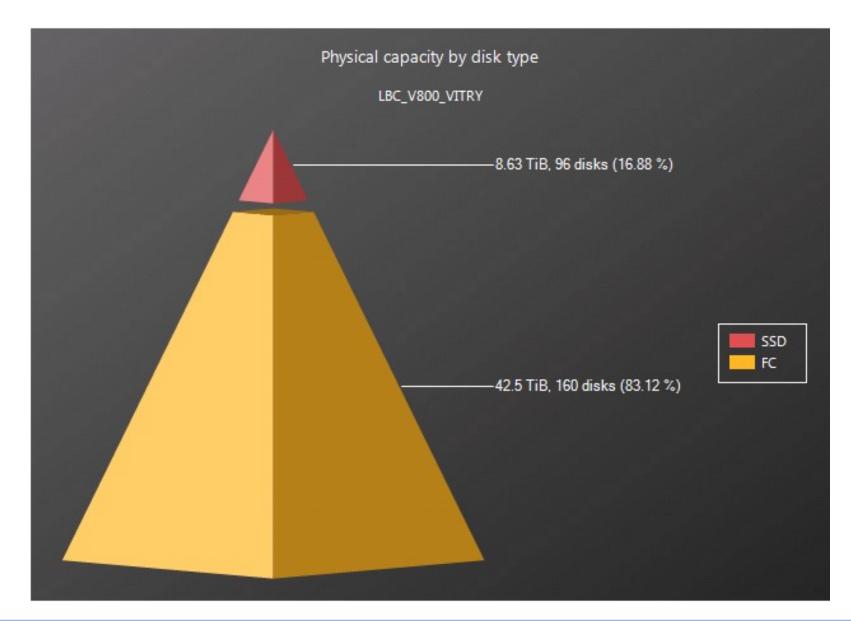
leboncoin.fr

Big Iron: 3Par V800 SAN



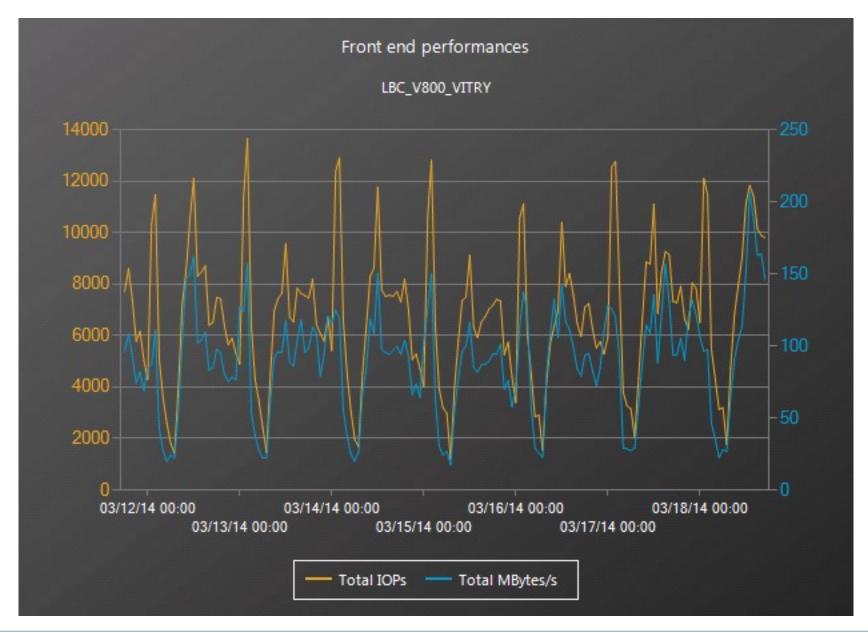


Big Iron: thin provisioning with mix of SSD and FC disks





Big Iron: high performance...



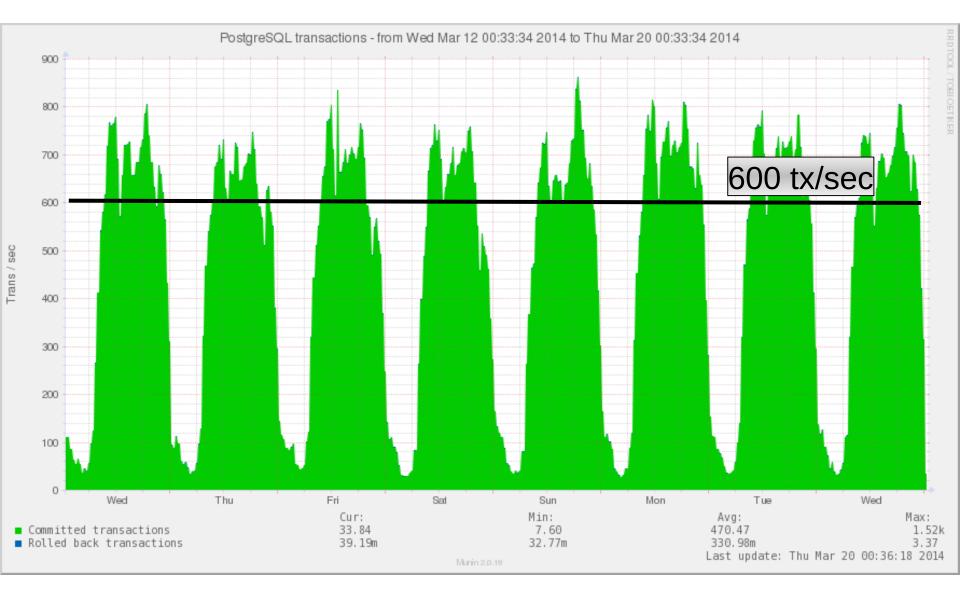


Bragging about it ;)



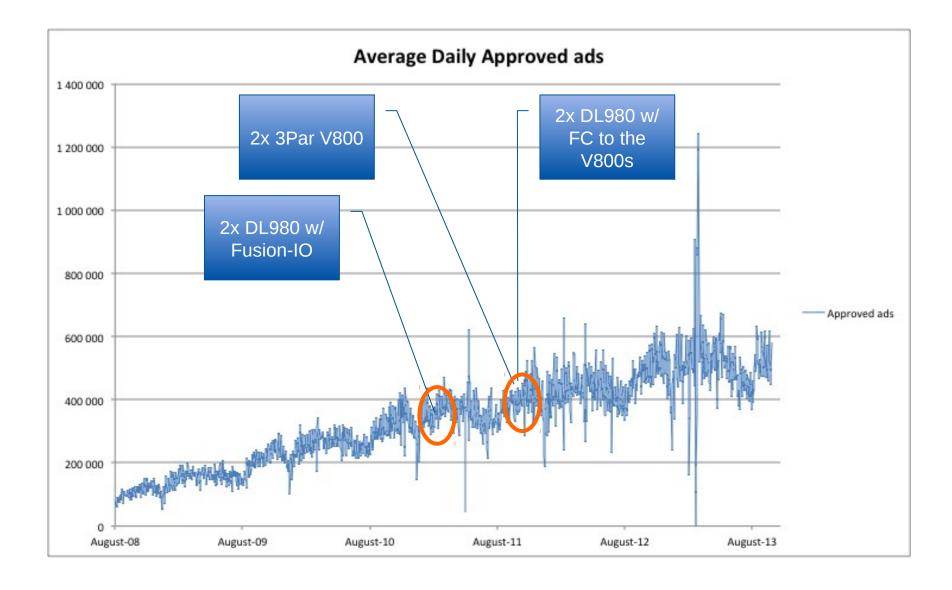


Despite the caches and the search engine, we get impressive workloads on the master DB



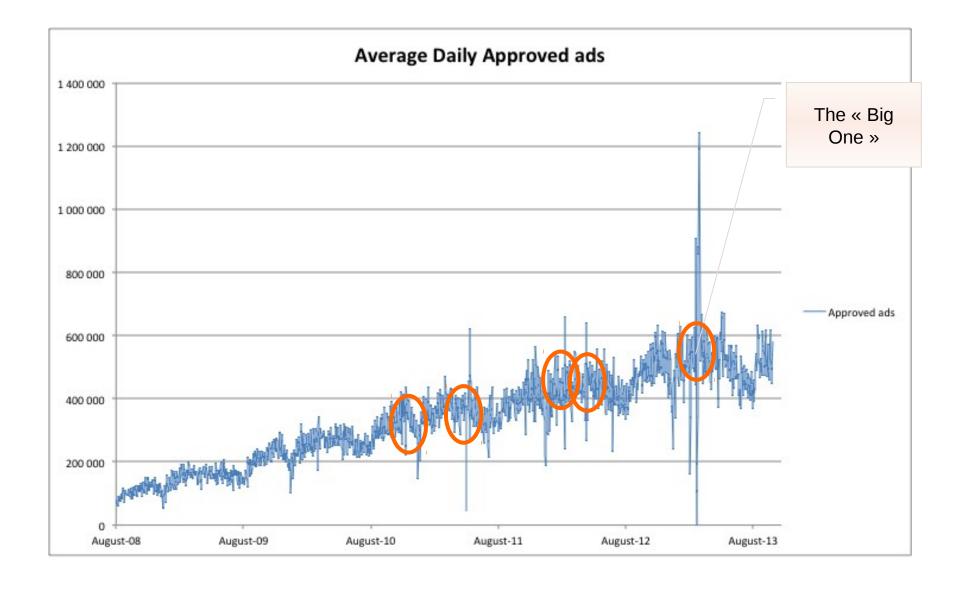


How did we get there?





We did go through growing pains and near disasters





Our own « worst day of our lives »: March 1st 2013 (1/2)

Master DB is slowing down dramatically

We find that Slony replication is the culprit

We don't know what to do...

Until we find a solution on the net that involves cleaning up slony metadata...

...(you know where this is going)...

We fumble. We notice. The slave is borked.

Rebuilding the Slave with slony brings the Master down. Oh. God...

We take the Master off the stack, and start rebuilding the slave w/ Slony

...5 days later, we are done (!)...



Our own « worst day of our lives »: March 1st 2013 (2/2)

...but we are not out of the woods yet!...

Pent up demand is bringing the site down!

We decide to switch to native replication!

...but the network cards are maxed out by the replication data...

...triggering a kernel bug...

...(Murphy, could you please step out of the room?)...

We implement network card bonding, and start moving support tables off the main instance

...and we are done!





What's Next?

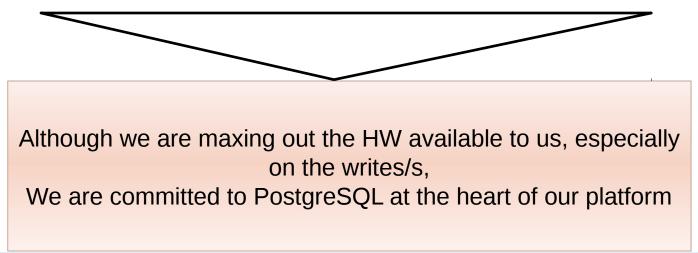
Vertical Scalability has limits!

We are already running on the biggest HW money can buy

Past certain volume levels, execution plans can change radically

Huge instances are difficult to maintain & backup safely

Rebuilding the slave in March 2013 took a full 5 days...





Key ideas to take our platform to the next level





Spread the reads & writes horizontally

Unbundling of

the schema



Move parts of the schema (that can be decoupled) to other instances. Spread the workload



Reduce application-level transaction interleaving by moving parts of the transactions to asynchronous workers



PGC

