

MySQL OS layer performance optimization Ten Tips

This article is mainly for some performance optimizations of mysql database on system layer.

Louis liu (vmcd)

DA@yihaodian

SHOUG Member

Twitter:@vmcd_gg

Ylouis83@gmail.com

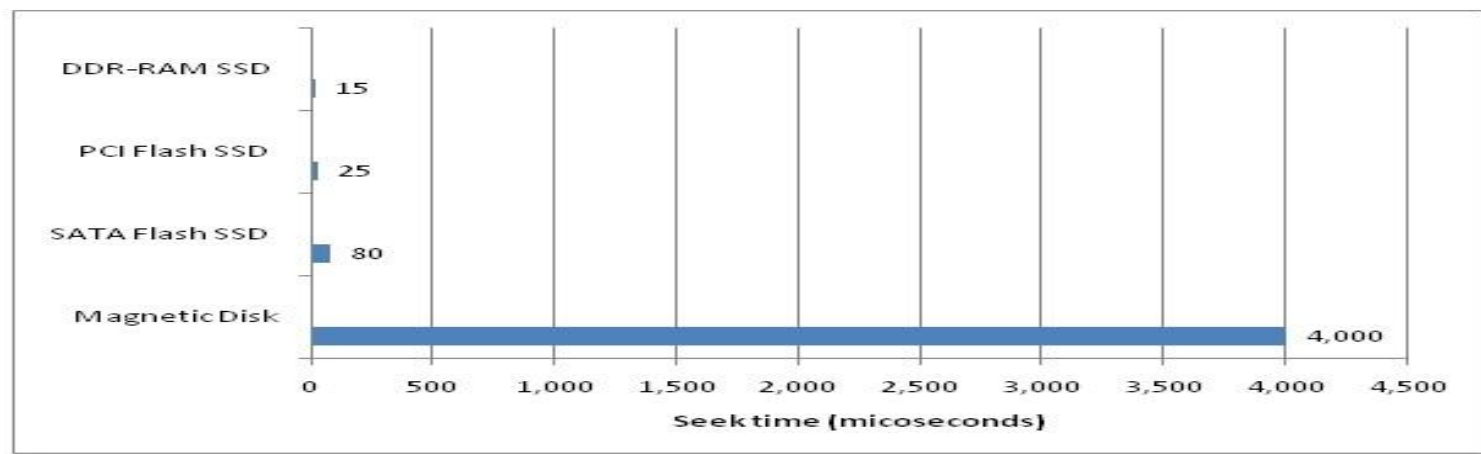
<http://www.vmcd.org> , <http://www.yhddba.com>

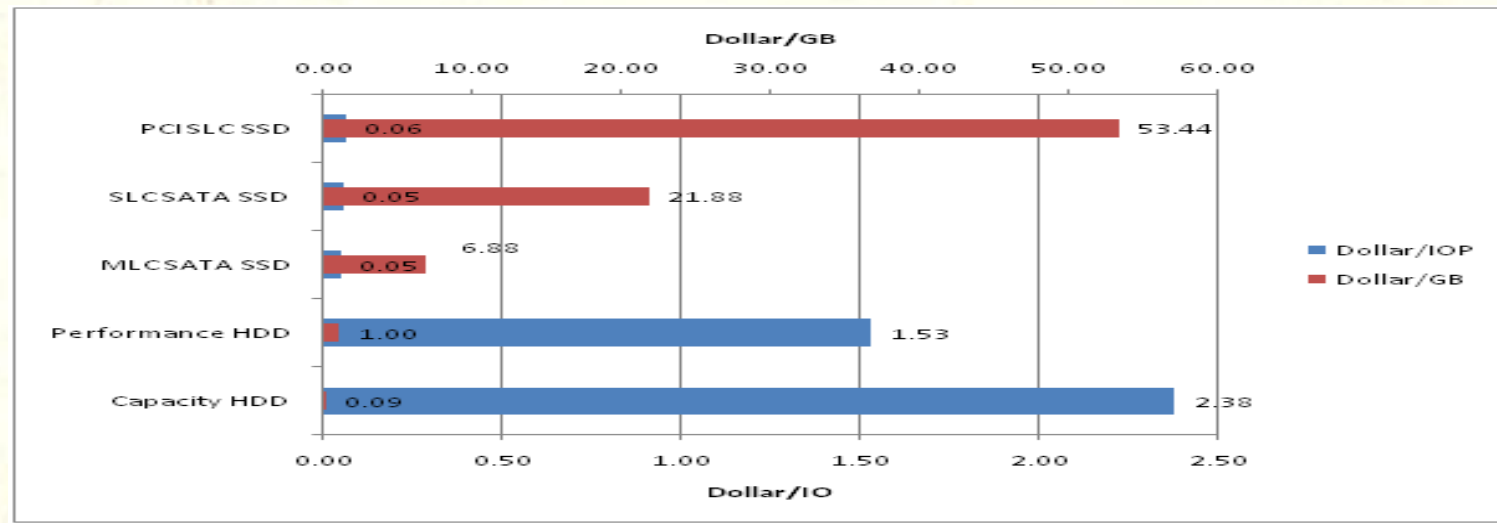
One: Disk tips

Normally we use SATA or SAS as database storage and sometimes the IOPS is enough. Today as High-growth businesses database becomes bigger and bigger and higher IOPS is require for OLTP system. (we first introduce SSD for some database system architect and now even PCIe like LSI or FIO for database storage)

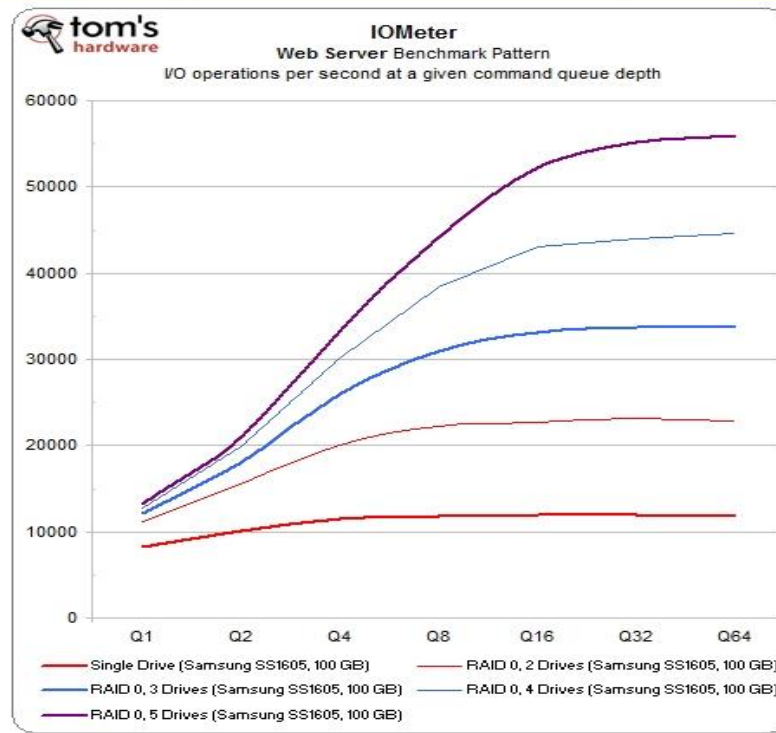
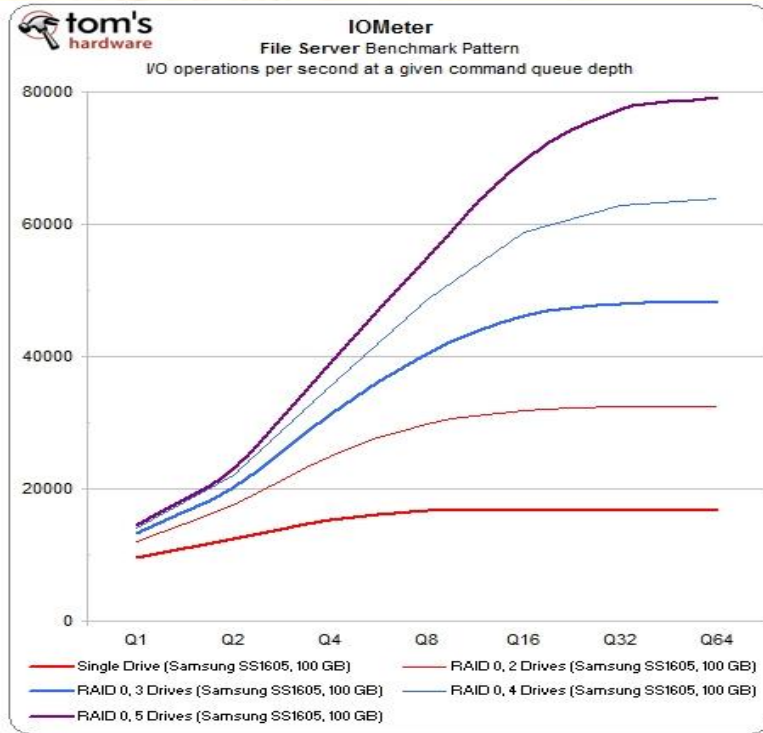
Some tips for using SSD and PCIe.

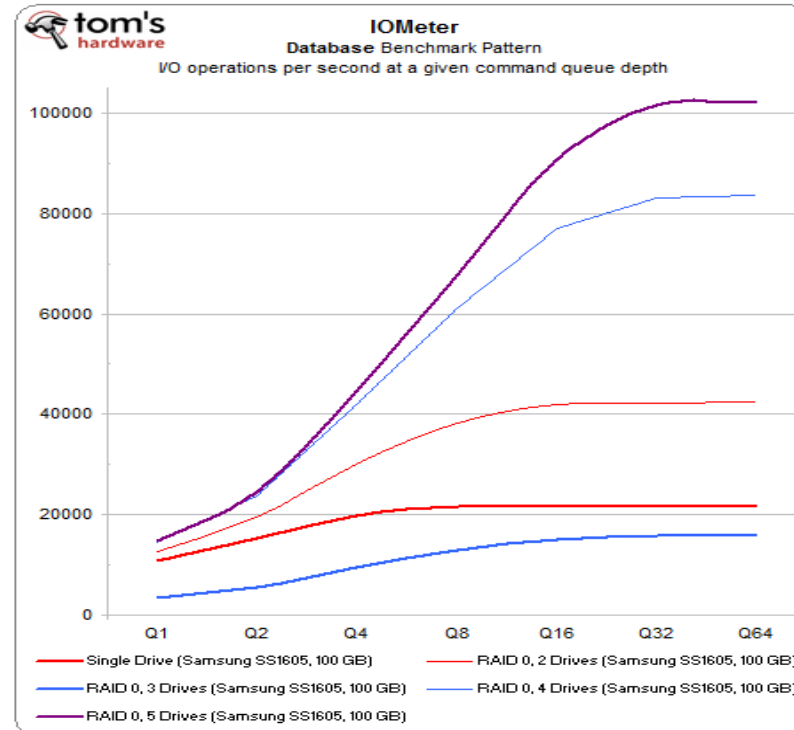
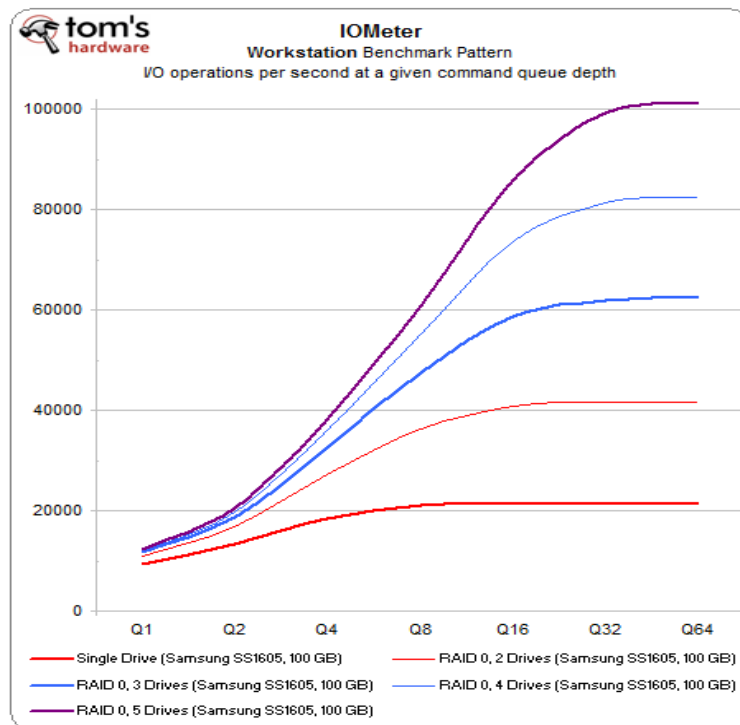
1. Regular SAS and SATA HDD : over 200 IOPS per driver, but SSD : over 2000+ IOPS (the new Intel SSD test benchmark even more than 10000 IOPS for random read and write), In our system design, we often use four SSD disks creating four database directories to store mysql data and we put sequence write file like binary log, db-logfile and some sequential binary file on SAS HDD because SSD is not good for sequential write. And we can even put some history and arch data on HDD disk as SAS HDD is cheaper than SSD and these data is not often used for reading.(Do H/W raid for HDD to get higher R/W IOPS)





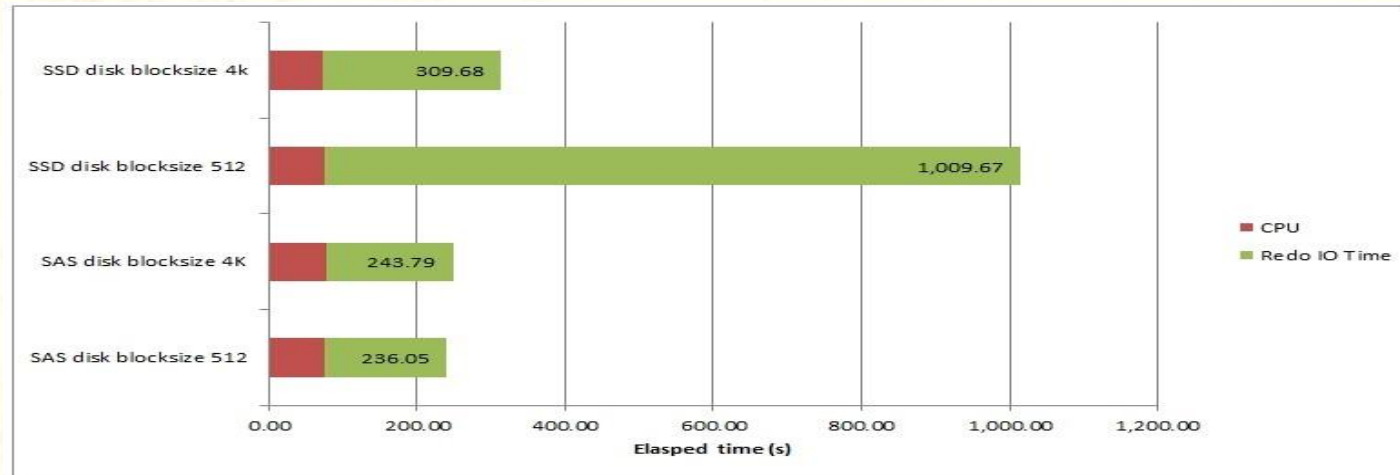
2. We don't make RAID (5/10/1) for SSD .Because of the lower write performance (we put most of reading operation on slave side. Although raid0 or raid10 can get better write performance but you need several SSDs to get it and that's not cheap!) . We use NO-RAID architect for mysql database and the HA design was done on OS layer (like we use M-S MHA DRBD M-M) to prevent mysql database single node of failure (for some core database we even create backup slave to do cold backup per day using VTL) and for PCIe you need to use S/W raid instead of H/W raid.





- SSD is not enough, memory is more important. With mysql 5.5 multi buffer pool was introduced so having a large buffer pool will get a nice IO performance. So don't save memory (that's very cheap 😊) we often use 128GB memory machine to run mysql database and even some core system the memory will be 256GB. As you know for mysql system even you have using SSD for storage, swap is bad, so give a large memory is very important.
- In our system, SSD is mainly for mysql database and PCIe is often given for oracle db. PCIe is very expensive but even better IOPS than SSD. We use large numbers of FIO cards to support oracle OLTP system. In our test - sequential write is also poor on PCIe (like oracle logfile),

oracle archive log) (In pic-1-1 test by Harrison, we must change log file block size to get better performance for sequential write on Flash-SSD)



[Pic-1-1](#)

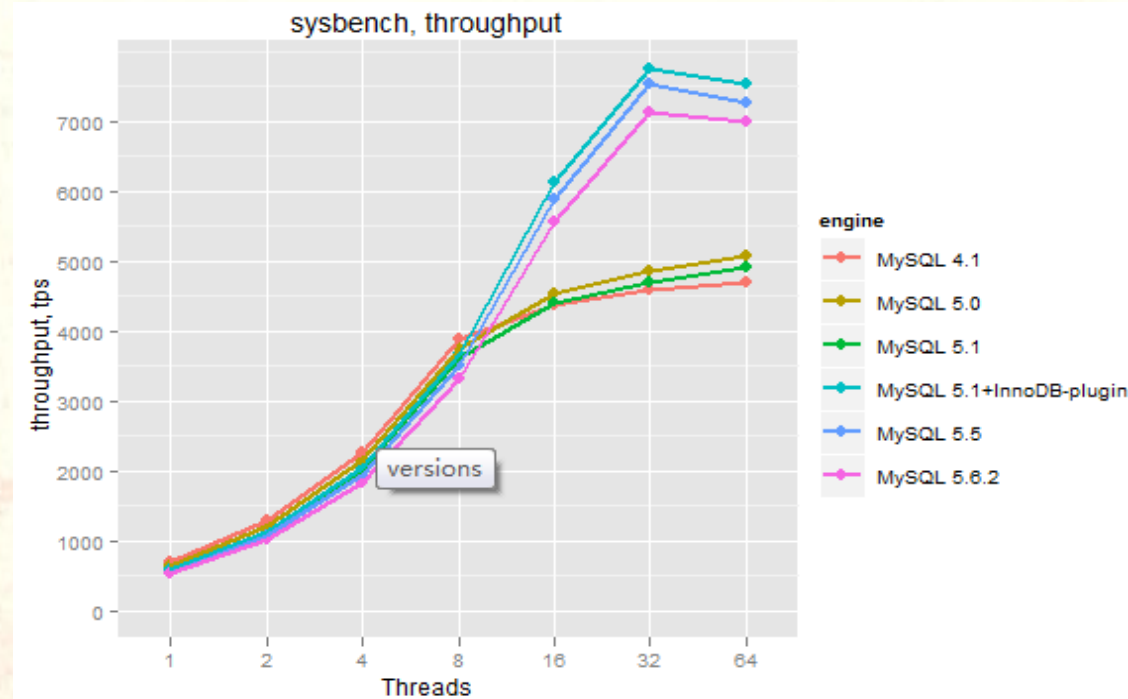
So remember that using large memory for important system (cheaper), using SSD for better IO performance (expensive) and do some data integration (history arch data on HDD drivers or even put different table on different disk drivers).

If you want more information for SSD and PCIe check this PDF

<http://www.slideshare.net/ylouis83/ssd-gc-review>

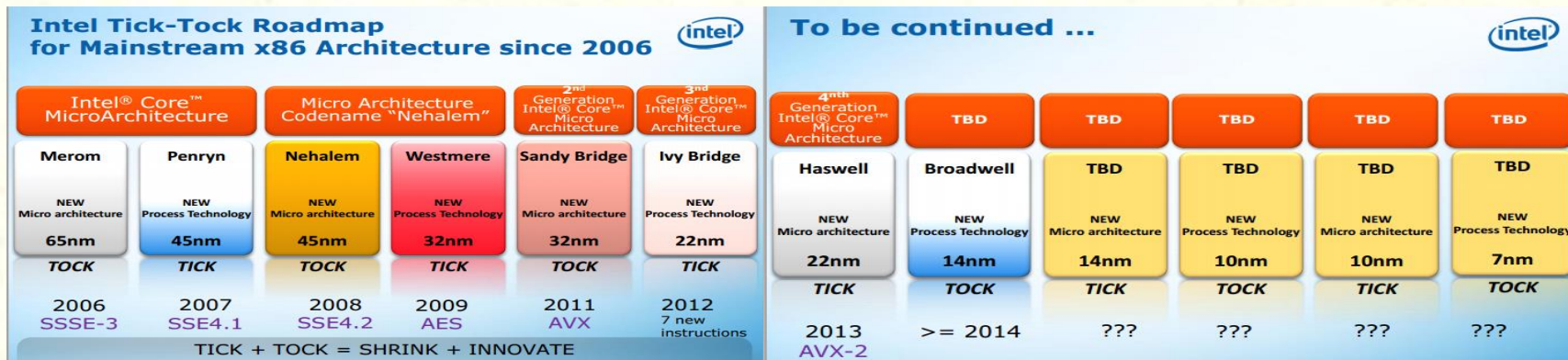
Two: DB version tips

This is very important for some small company. Many of them even use very old mysql version so there will be a very big problem for giving a better performance they want. As you now, too old mysql version have many problems (lower performance on SMP and large memory system) and even some lock and mutex problems. From mysql 5.1 to mysql 5.5 we got performance promotion a lot and even in 5.6 and 5.7 version performance of benchmark is still more excellent (visit Dimitrik blog for more information)



<http://dimitrik.free.fr/blog/archives/2013/11/mysql-performance-over-1m-qps-with-innodb-memcached-plugin-in-mysql-57.html>

Today we use new Nehalem and Sandy bridge CPU architect (I don't think using an very old machine with even linux4.x to run database system is a good choice) so please upgrade your DB version (5.1 to 5.5 is very important even 5.6 for better performance and a lot of new feature to help design your DB system)



```

processor      : 157
vendor_id     : GenuineIntel
cpu family    : 6
model         : 47
model name    : Intel(R) Xeon(R) CPU E7- 8870 @ 2.40GHz
stepping     : 2
cpu MHz       : 1064.000
cache size   : 30720 KB
physical id   : 7
siblings     : 20
core id      : 7
cpu cores    : 10
apicid       : 239
fpu          : yes
fpu_exception : yes
cpuid level  : 11
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr p
arat_pni_monitor ds cpl vmx smx est tm2 sse3 cx16 xtpr sse4_1 sse4
bogomips     : 4788.06
clflush size : 64
cache alignment : 64
address sizes : 44 bits physical, 48 bits virtual
power management: [8]

processor      : 63
vendor_id     : GenuineIntel
cpu family    : 6
model         : 45
model name    : Intel(R) Xeon(R) CPU E5-4620 0 @ 2.20GHz
stepping     : 7
cpu MHz       : 1200.000
cache size   : 16384 KB
physical id   : 3
siblings     : 16
core id      : 7
cpu cores    : 8
apicid       : 111
fpu          : yes
fpu_exception : yes
cpuid level  : 13
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr p
arat_pni_monitor ds cpl vmx smx est tm2 sse3 cx16 xtpr sse4_1 sse4
bogomips     : 4399.97
clflush size : 64
cache alignment : 64
address sizes : 46 bits physical, 48 bits virtual
power management: [8]
    
```

Tips: look at these pictures you see actual CPU MHZ is not the value I have marked red, these CPUs are in saving mode so machine's full power will not be played out. (but in these two machines saving mode maybe enough as total 160 and 64 lcpu 😊)

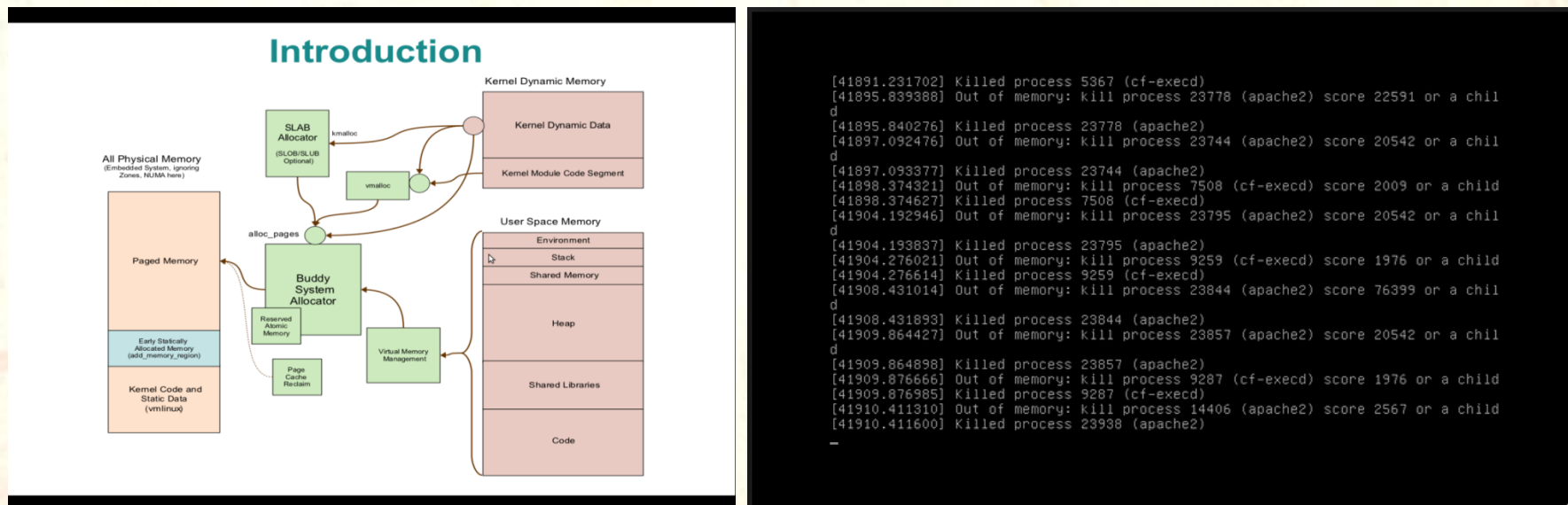
Three: avoid Swap in your DB system

As I said before in this article, memory is very cheap now so don't save your money on memory. If your database system has a lot of swap I can make sure your database is very slow. And how to determine the size of buffer pool ? Just keep "hot data" should be cached in memory , that's why we say if you system has many FTS query your system must very slow.

Swap setting tips

1. do not set swap size to zero

If you set swap size to zero OOM kill may happen (maybe kill mysqld for free memory for system)



We can set "oom_kill_allocating_task" , "/proc/&PID/oom_adj" and "vm.min_free_kbytes" to avoid OOM killer but set swap size to zero is still very dangerous.

2. set "vm.swappiness=0"

Set this parameter to zero to ensure mysql can hold hot page for long time

3. Some other drivers can cause swap

Be careful when using some Native drivers like fusion-IO (sometimes when disk recovering system memory should be used) and design your database memory allocation (not use fully memory like 128GB OS memory: 100GB for buffer pool) monitor your system all the time if swap happens you must check your application to avoid OOM killer.

➤ **Advantages**

- Fast time to market and production
- Little chip design NRE needed
- Field upgradeable
- Custom algorithms (H/W optimized)

➤ **Disadvantages**

- High FPGA power consumption limits IOPS
- Likely to be highest device cost (FPGA)

**Example Implementation:
FPGA-based
FUSION-io ioDrive2 SSC**



FPGA based HBA + Controller

- Up to 535,000 IOPs (512b, MLC)
- 24W
- 1.28TB

Higher IOPS/power SSCs also available

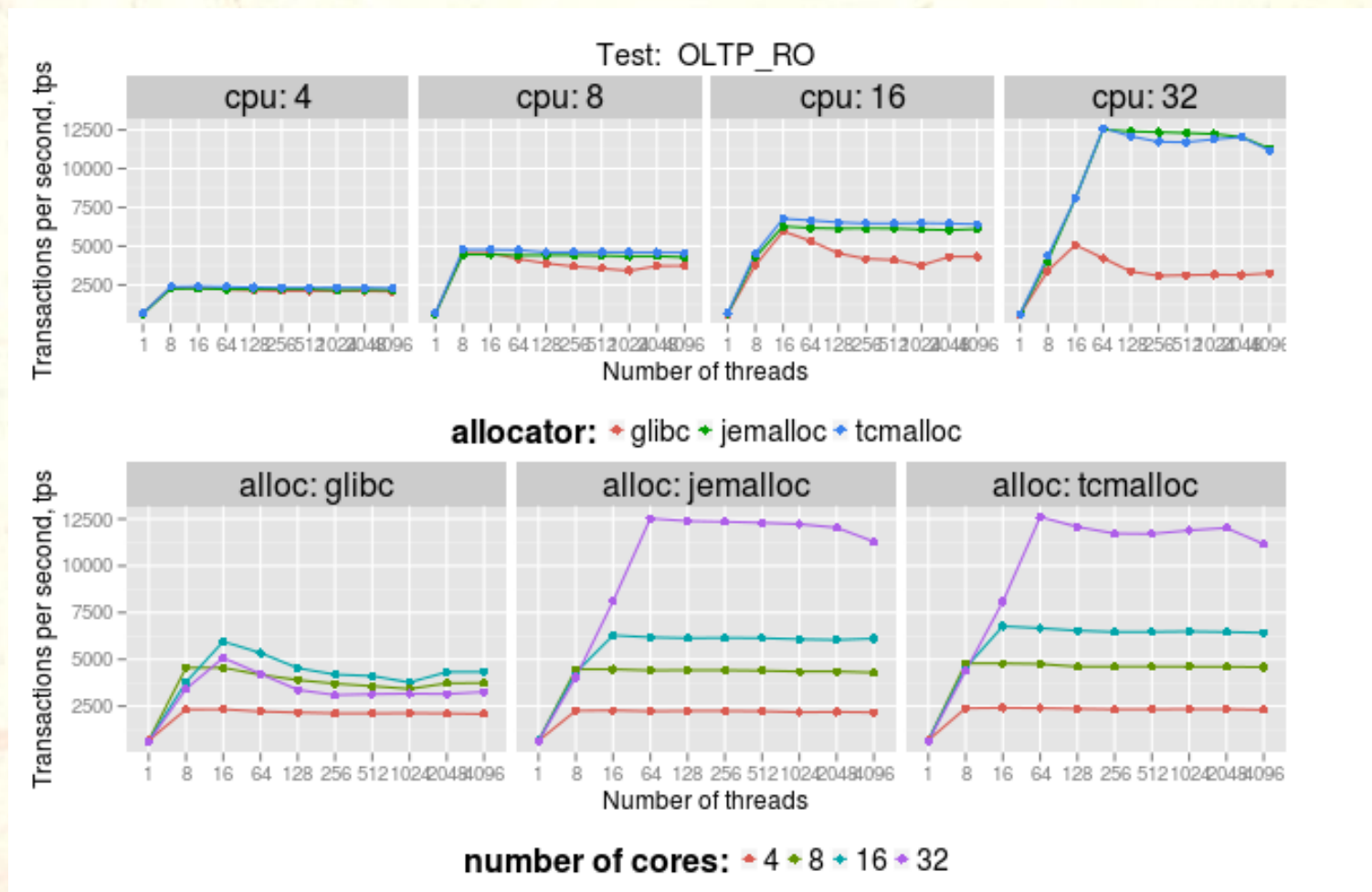
Note: Data taken from Fusion-io website

memory allocator	mysqld RSS size grow(kbytes)	mysqld VSZ size grow(kbytes)
lockless	6.966.736	105.780.880
jemalloc-2.2.5	214.408	3.706.880
jemalloc-3.0	216.084	5.804.032
tcmalloc	456.028	514.544
glibc-2.13-new-malloc	210.120	232.624
glibc-2.13-old-malloc	253.568	1.006.204
glibc-2.12.1-system	162.952	215.064
glibc-2.15-new-malloc	5.106.124	261.636

Facebook testing jemalloc blog:

<http://www.facebook.com/notes/mysql-at-facebook/using-jemalloc-to-fix-a-performance-problem/10150494400690933>

http://www.reddit.com/r/programming/comments/18zija/github_get_30_better_performance_using_tcmalloc/



In our future planning, we're still thinking using tcmalloc for the first choice of memory allocator (we consider to use tcmalloc instead of

default on new mysql 5.6 database)

Five : file system options and IO management

For file system mount options (ext3 ext4 xfs and so on) disable barrier is a good choice.

Like : “mount -o barrier=0” on ext3

Show mount options :

```
[root@db-3-27 ~]$ mount | grep 'data'  
/dev/sdb1 on /data type ext4 (rw,noatime,nodiratime,nobarrier)
```

MS_NOATIME

Do not update access times for (all types of) files on this filesystem.

MS_NODIRATIME

Do not update access times for directories on this file system.





This flag provides a subset of the functionality provided by MS_NOATIME; that is, MS_NOATIME implies MS_NODIRATIME.

```
void touch_atime(struct vfsmount *mnt, struct dentry *dentry)
{
    /* ... */
    if (inode->i_flags & S_NOATIME)
        return;
    if (IS_NOATIME(inode))
        return;
    if ((inode->i_sb->s_flags & MS_NODIRATIME) && S_ISDIR(inode->i_mode))
        return;
```

noatime is a superset of nodiratime

The main ones are replacing atime/relatime with noatime. This causes the FS to not write read-times to a file when read. Think about it.

Enable BBWC on RAID card is important .write cache will give a huge promotion on performance, but write cache must be protected by battery so battery's discharge and recharge will be problem, in that time BBWC will set cache policy from write back to write through so at this time db system's IO will be slow. This problem was resolved by Flash like USB.—FBWC (Flash-Based Write Cache) copy data in cache to flash and then rewrite to HDD.

Technical specifications	HP Smart Array P212	HP Smart Array P410	HP Smart Array P411	HP Smart Array P812
				
Data compatibility	Yes, future SAS Smart Array controllers	Yes, future SAS Smart Array controllers	Yes, future SAS Smart Array controllers	Yes, future SAS Smart Array controllers
Instant upgrades to other Smart Array controllers	Yes, SAS Smart Array controllers	Yes, SAS Smart Array controllers	Yes, SAS Smart Array controllers	Yes, SAS Smart Array controllers
Consistent software management tools	Yes	Yes	Yes	Yes
PCI bus	PCIe 2.0 x8	PCIe 2.0 x8	PCIe 2.0 x8	PCIe 2.0 x8
PCI peak data transfer rate	2Gb/s	2Gb/s	2Gb/s	4Gb/s
SCSI protocol support	6Gb/s SAS/3Gb/s SATA	6Gb/s SAS/3Gb/s SATA	6Gb/s SAS/3Gb/s SATA	6Gb/s SAS/3Gb/s SATA
SCSI peak data transfer rate	6Gb/s per port	6Gb/s per port	6Gb/s per port	6Gb/s per port
Channels	8 SAS ports	8 SAS ports	8 SAS ports	24 SAS ports
Connectors (External/Internal)	1/1	0/2	2/0	4/2
Drives supported (Max)	32	64	64	108
Cache	0MB RAM, with available 256MB and BBWC upgrade	256MB DDR2- RAM, with available BBWC upgrade,	256MB DDR2- RAM, with available BBWC upgrade,	1G FBWC

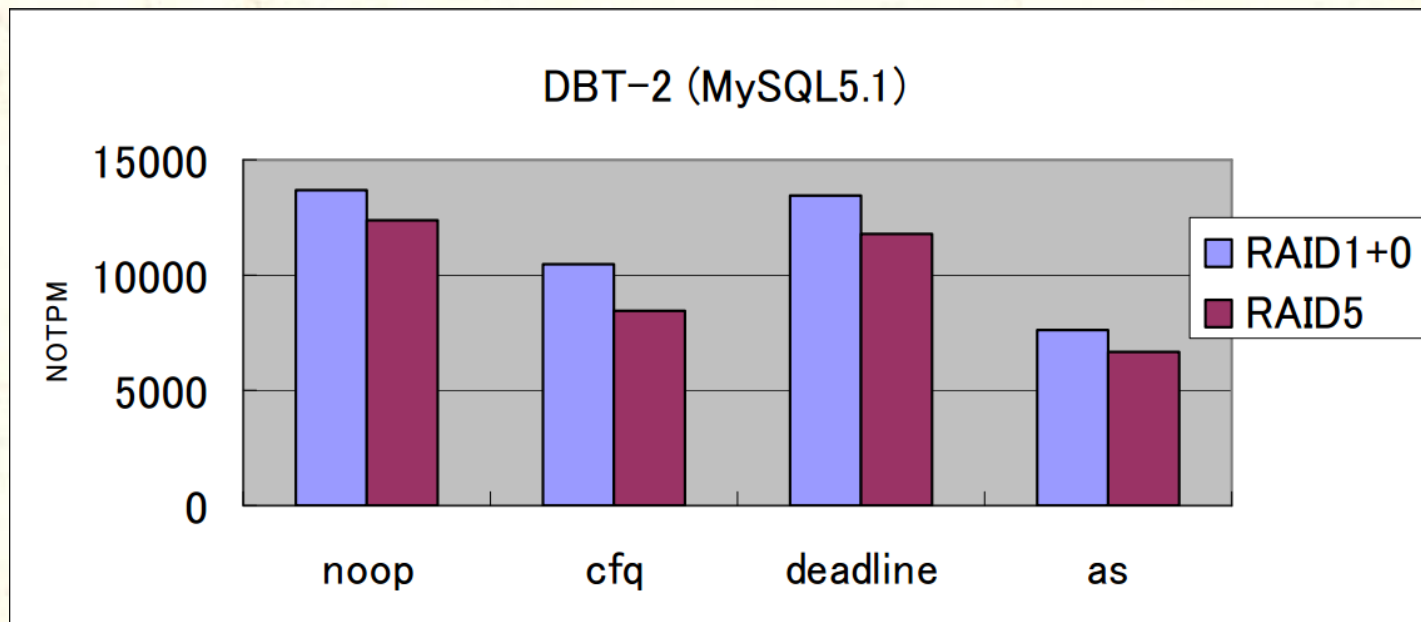
For battery recharge problem we have some solutions:

Discharge and recharge manual you can do these in midnight.

Set cache policy to “force write back” but you may lost data.

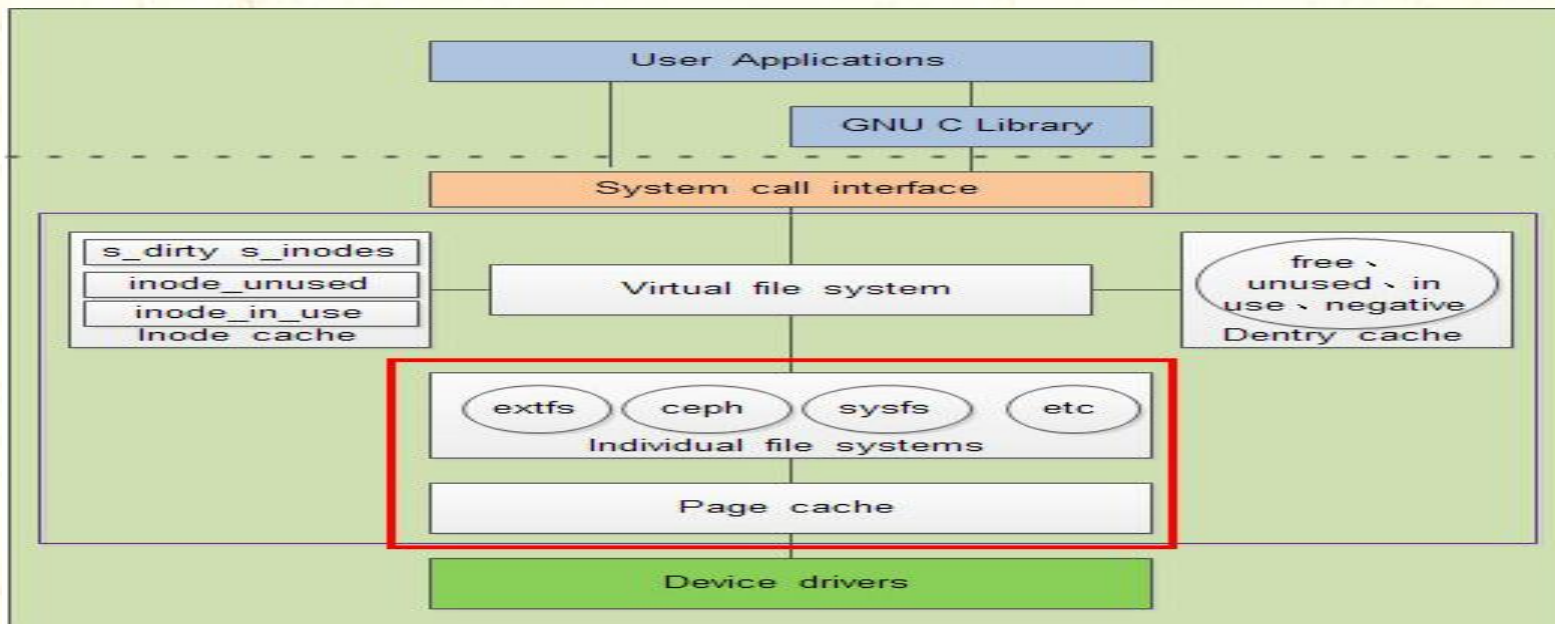
For RAID card policy we can disable read cache (direct read) to let write cache use total cache. And we can also disable prepare read.

And if you use SSD for storage you can enable Fastpath feature in LSI raid card to get better performance. Using PCIe like fusion-IO you can enable NOOP to get better IO performance.



Set "innodb_flush_method= O_DIRECT"

Avoiding talking with OS cache, directly write file, and using Fsync() to flush dirty cache to disk.(flush including Inode cache,Buffer cache,Directory cache)



	Open log	Flush log	Open datafile	Flush data
Fdatasync		fsync()		fsync()
O_DSYNC	O_SYNC			fsync()
O_DIRECT		fsync()	O_DIRECT	Fsync()

Six : using huge page to get good performance

Applications that perform a lot of memory accesses (several GBs) may obtain performance improvements by using large pages due to reduced Translation Lookaside Buffer (TLB) misses. HugeTLBfs is memory management feature offered in Linux kernel, which is valuable for applications that use a large virtual address space. It is especially useful for database applications such as MySQL, Oracle and others. Other server software that uses the prefork or similar (e.g. Apache web server) model will also benefit.

25% - 300% improvement: *"These memory accesses are then frequently cache misses which introduces a high latency to the memory request. Increasing page sizes from 4K to 16M significantly reduces this problem as the number of tlb misses drops. Typically it will reduce runtimes by 25-30% but in an extreme case I've seen an SPH code run 3x faster simply by enabling large pages."*

You can config by this article:

<http://www.cyberciti.biz/tips/linux-hugetlbfs-and-mysql-performance.html>

Seven : Running mysql on Numa architect system

Today we also run multi mysql instance on numa architect system so nice design will be very important.

Mysql's cpu utilization is always a problem (still not very good as oracle today) so when run mysql on multi core machine we can manual control resource usage for multi instances to get better performance (we have two IBM x3950 machines , and each of them has 160 lcpu with Hyper-threading enabled, so on this type machine only one mysql instance is too waste)

Not only mysql but also some other database even Nosql like mongodb , Numa architect still has variable problems when use. In oracle database DBAs often be suggested to disable numa feature as oracle often just run one instance per machines and also for mongodb.

Running MongoDB on a system with Non-Uniform Access Memory (NUMA) can cause a number of operational problems, including slow performance for periods of time or high system process usage.

When running MongoDB on NUMA hardware, you should disable NUMA for MongoDB and instead set an interleave memory policy.

Note MongoDB version 2.0 and greater checks these settings on start up when deployed on a Linux-based system, and prints a warning if the system is NUMA-based.

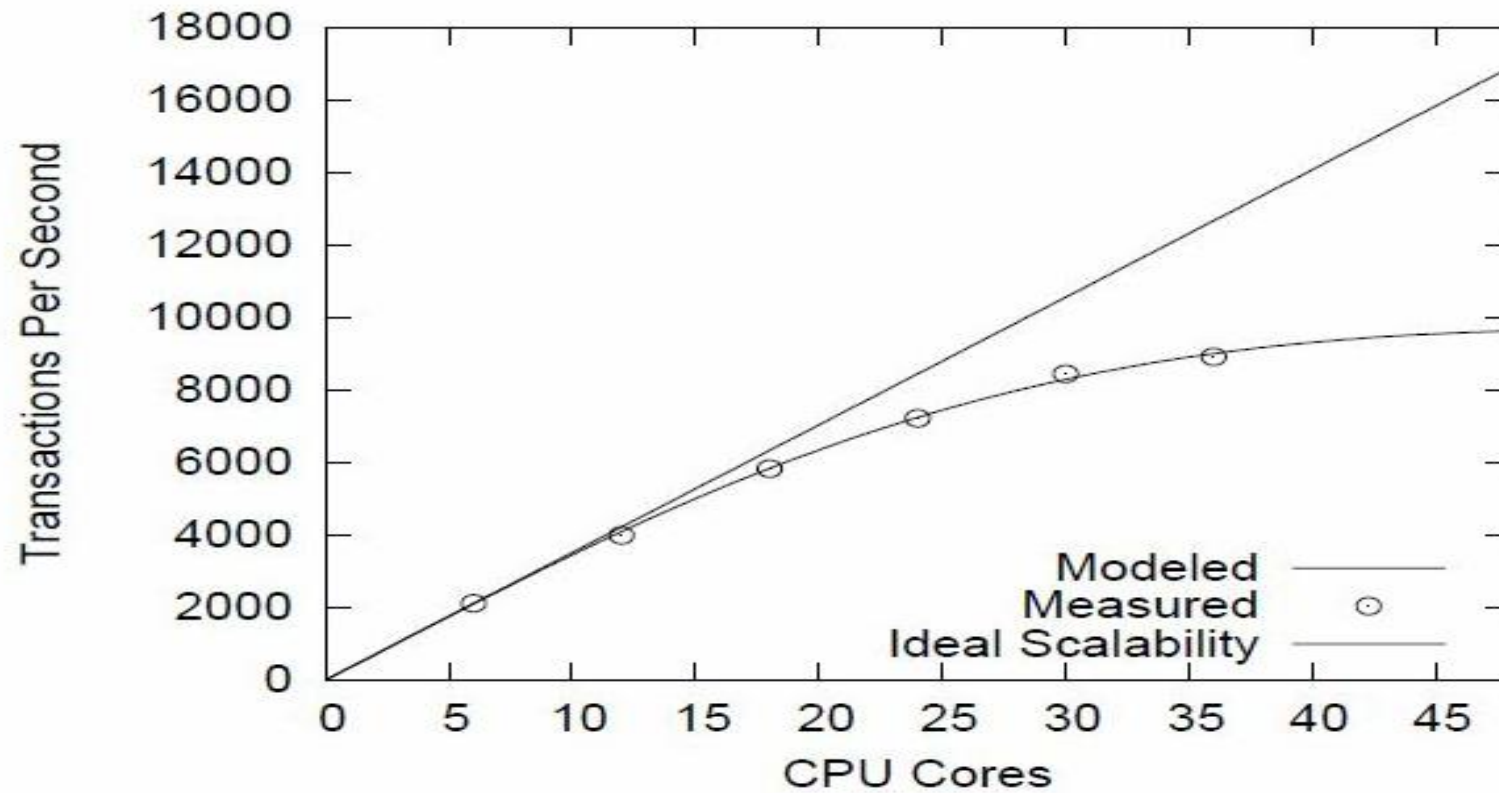
To disable NUMA for MongoDB and set an interleave memory policy, use the numactl command and start mongod in the following manner:

```
numactl --interleave=all /usr/bin/local/mongod
```

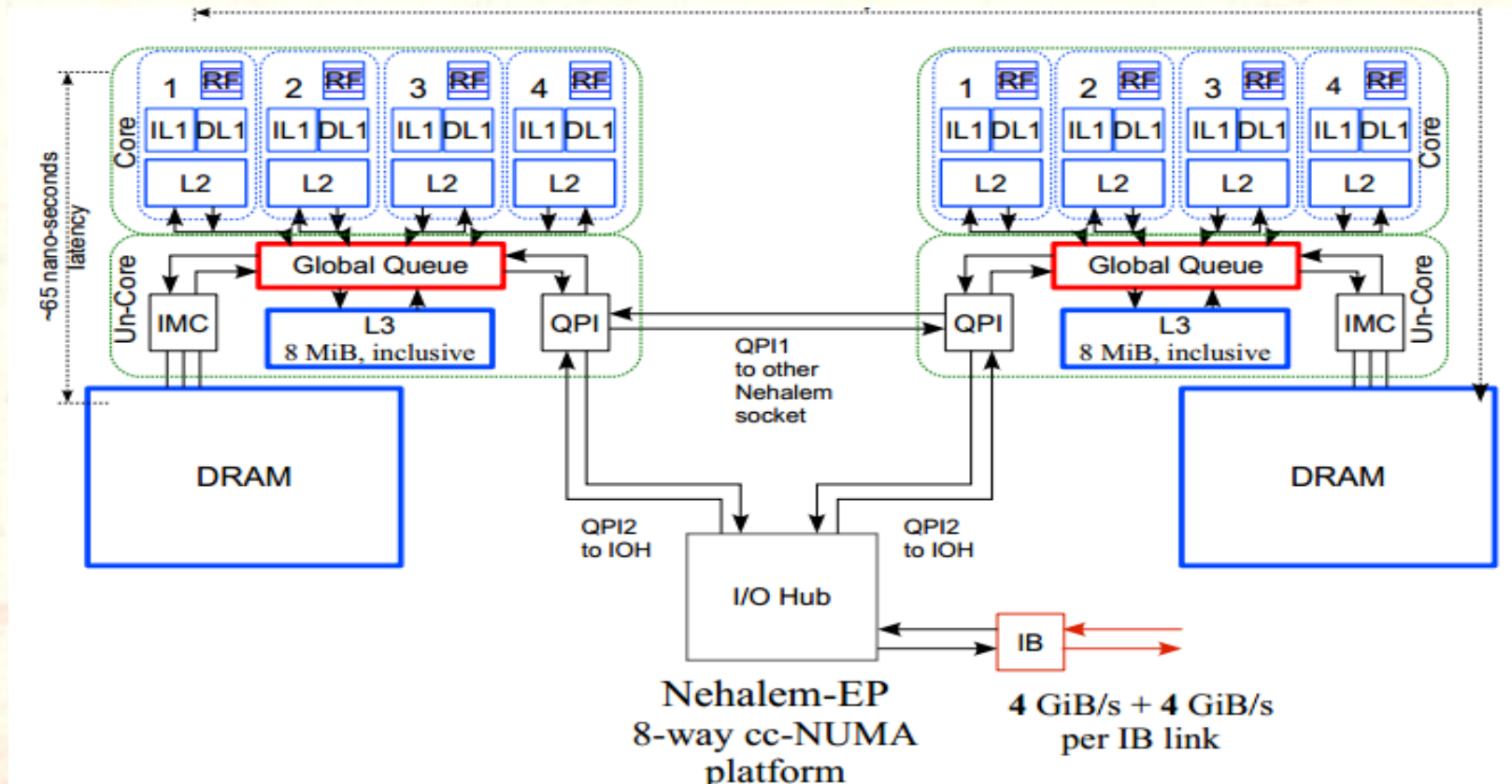
Then, disable zone reclaim in the proc settings using the following command:

```
echo 0 > /proc/sys/vm/zone_reclaim_mode
```


See blew picture (percona test on mysql 5.1)



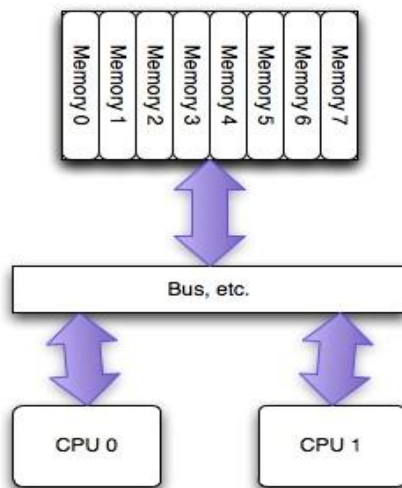
NUMA architect pic (for Intel NUMA)



General NUMA architect

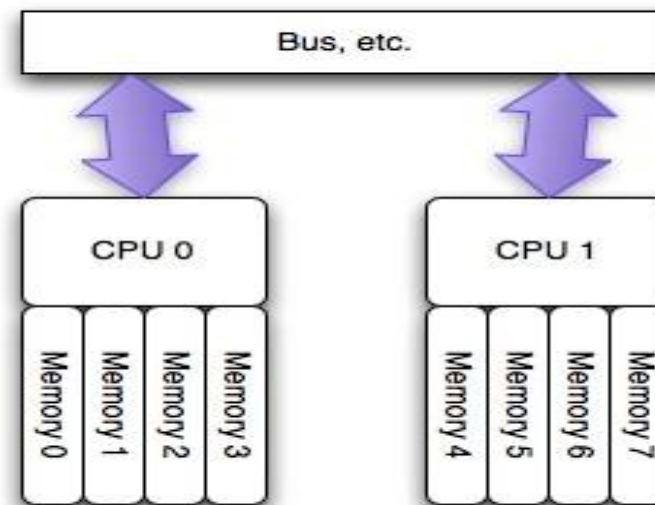
Contrasting the SMP/UMA and NUMA architectures

The SMP/UMA architecture



The SMP, or UMA architecture, simplified

The NUMA architecture



The NUMA architecture, simplified

Allocate memory with a particular policy:

- * locally on the "current" node — using `--localalloc`, and also the default mode
- * preferably on a particular node, but elsewhere if necessary — using `--preferred=node`

- * always on a particular node or set of nodes — using `--membind=nodes`
- * interleaved, that is, spread evenly round-robin across all or a set of nodes — using `--interleaved=all` or `--interleaved=nodes`

Run the program on a particular node or set of nodes, in this case that means physical CPUs (`--cpunodebind=nodes`) or on a particular core or set of cores (`--physcpubind=cpus`).

You see the default policy locally on the “current” node may cause “swap insanity”.

To avoid this problem (if only one instance we can disable NUMA) we should do some resource limitation

1. CPU and Memory

Using `numactl -cpubind=x --localalloc` to bind instance to different CPUs
or using `taskset`:

OPTIONS

`-p, --pid`

operate on an existing PID and not launch a new task

`-c, --cpu-list`

specify a numerical list of processors instead of a bitmask.
The list may contain multiple items, separated by comma, and ranges. For example, `0,5,7,9-11`.

Eg : `# taskset -pc 0,12,2,64 /*your mysql pid*/`

2. IO

We consider use IO resource management to manage IO usage by different Instance (In linux 6 often use cgroups to manage IO)

Eg:

```
group mysql_admin {
  cpuset {
    cpuset.cpus = "1,2,3,4,5,6,7,8";
    cpuset.mems="0";
  }
  memory {
    memory.limit_in_bytes=1086778;
    memory.memsw.limit_in_bytes=108657800;
    memory.swappiness=0;
  }
  blkio {
    blkio.throttle.read_iops_device = "5:0 1000";
  }
}
```

3. Network

Using different IP address for different instance (per instance has it's own bonding-IP address) so we need many NICs.

4. Network advance

Even bind different network interrupt (network bottlenecks, Hardware interrupts) on different CPUs

```
[root@ITEM-LGSTD01 ~]# mpstat -P ALL
Linux 2.6.32-300.10.1.el5uek (ITEM-LGSTD01) 01/13/2014

12:40:43 AM CPU %user %nice %sys %iowait %irq %soft %steal %idle intr/s
12:40:43 AM all 12.79 0.00 2.48 0.11 0.00 0.23 0.00 84.38 21326.99
12:40:43 AM 0 18.96 0.00 3.51 0.09 0.00 0.46 0.00 76.98 0.00
12:40:43 AM 1 18.73 0.00 3.21 0.23 0.00 0.42 0.00 77.40 0.00
12:40:43 AM 2 6.23 0.00 1.14 0.10 0.00 0.00 0.00 92.54 0.00
12:40:43 AM 3 14.95 0.00 2.48 0.03 0.00 0.27 0.00 82.28 0.00
12:40:43 AM 4 15.36 0.00 2.72 0.02 0.00 0.28 0.00 81.62 0.00
12:40:43 AM 5 2.55 0.00 0.43 0.03 0.00 0.00 0.00 97.00 0.00
12:40:43 AM 6 19.56 0.00 3.85 0.17 0.00 0.38 0.00 76.03 0.00
12:40:43 AM 7 18.25 0.00 3.79 0.07 0.00 0.36 0.00 77.53 0.00
12:40:43 AM 8 16.97 0.00 3.52 0.05 0.00 0.32 0.00 79.14 0.00
12:40:43 AM 9 16.63 0.00 3.26 0.03 0.00 0.30 0.00 79.77 0.00
12:40:43 AM 10 3.13 0.00 0.82 0.03 0.00 0.00 0.00 96.02 0.00
12:40:43 AM 11 2.15 0.00 0.44 0.02 0.00 0.00 0.00 97.39 0.00
12:40:43 AM 12 17.76 0.00 4.29 0.17 0.00 0.35 0.00 77.41 0.00
12:40:43 AM 13 17.87 0.00 3.73 0.11 0.00 0.35 0.00 77.94 0.00
12:40:43 AM 14 17.84 0.00 4.09 0.83 0.00 0.41 0.00 76.83 0.00
12:40:43 AM 15 17.60 0.00 3.23 0.09 0.00 0.34 0.00 78.74 0.00
12:40:43 AM 16 3.29 0.00 0.78 0.12 0.00 0.00 0.00 95.81 0.00
12:40:43 AM 17 2.29 0.00 0.49 0.12 0.00 0.00 0.00 97.11 0.00
12:40:43 AM 18 18.85 0.00 3.54 0.15 0.00 0.35 0.00 77.10 0.00
12:40:43 AM 19 18.89 0.00 3.30 0.06 0.00 0.35 0.00 77.40 0.00
12:40:43 AM 20 17.06 0.00 2.89 0.04 0.00 0.30 0.00 79.70 0.00
12:40:43 AM 21 16.39 0.00 2.99 0.11 0.00 0.30 0.00 80.20 0.00
12:40:43 AM 22 3.08 0.00 0.58 0.03 0.00 0.00 0.00 96.30 0.00
12:40:43 AM 23 2.10 0.00 0.34 0.02 0.00 0.00 0.00 97.54 0.00
12:40:43 AM 24 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
12:40:43 AM 25 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

```
[BEGIN] 2014/1/13 0:42:45
cat /proc/interrupts

0: 0 CPU16 CPU17 CPU18 CPU19 CPU20 CPU21 CPU22 CPU23 IO-APIC-edge timer
4: 0 0 0 0 0 0 0 0 IO-APIC-edge
8: 0 0 0 0 0 0 0 0 IO-APIC-edge rtc0
9: 0 0 0 0 0 0 0 0 IO-APIC-fasteoi acpi
16: 0 0 0 0 0 0 0 0 IO-APIC-fasteoi ehci_hcd:usb1, ehci_hcd:usb2
39: 0 0 0 0 0 0 0 0 PCI-MSI-edge megasas
40: 0 0 0 0 0 0 0 0 PCI-MSI-edge iodrive-0000:58:00.0-0
59: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2
60: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-0
61: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-1
62: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-2
63: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-3
64: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-4
65: 0 0 0 0 2957290671 0 0 0 PCI-MSI-edge eth2-TxRx-5
66: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-6
67: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth2-TxRx-7
68: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3
69: 0 0 0 2929089167 0 0 0 0 PCI-MSI-edge eth3-TxRx-0
70: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-1
71: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-2
72: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-3
73: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-4
74: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-5
75: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-6
76: 0 0 0 0 0 0 0 0 PCI-MSI-edge eth3-TxRx-7
```

Eight: IDC replication consideration

In our system we have some remote IDC's replication requirement (like from shanghai IDC to Beijing IDC) mysql M-S apply maybe delay When meet some cases.

Some suggestion and solutions (we haven't deployed all of them ☺ just plan)

1. Use private network connection (for using 1GB private network connection) and we can do some policies for mysql replications.
2. Use advanced network technologies like MPLS or "Intel DDIO DPDK" (I'm not confirm this as we not used so far) and CPU must support DDIO

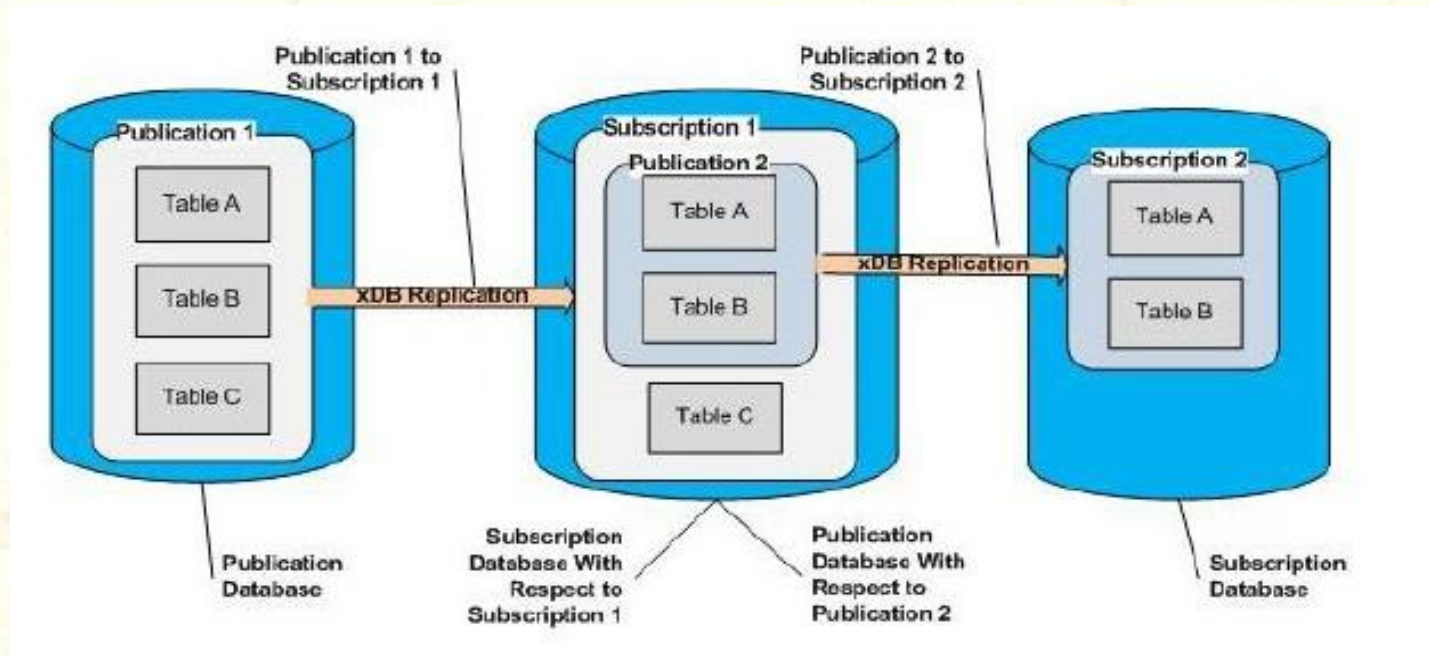


For more information:

<http://dpmk.org/doc>

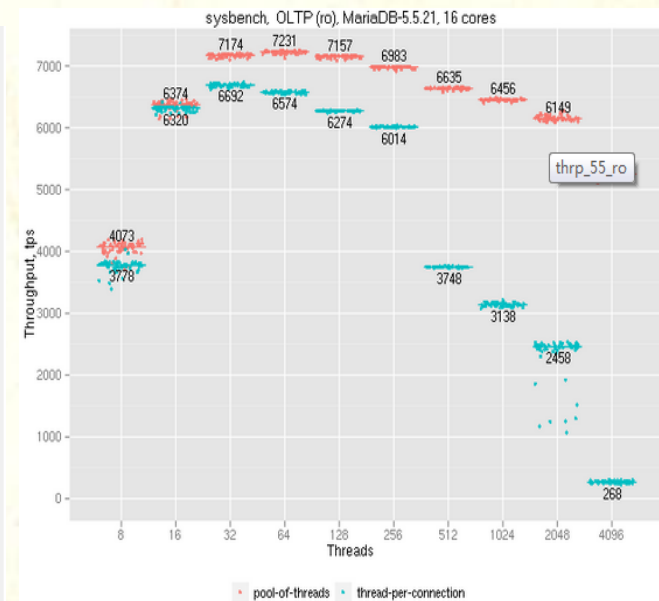
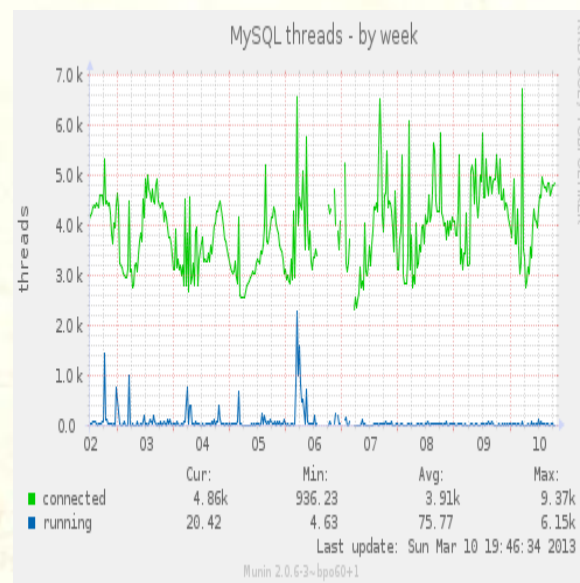
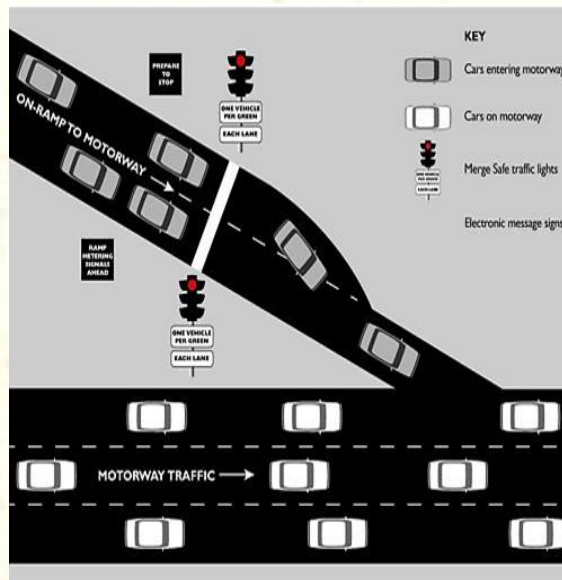
<http://www.intel.com/content/www/us/en/io/direct-data-i-o.html>

3. Use a cascade architect (from china to USA) we create database in HK as middle database and rep from China->HK->USA even can add Some policies you need.



Nine: Using Thread pool and HS

1. Thread pool was first introduced by Percona server (like Oracle multithreaded server using dispatcher to do some traffic control)



2. HS (HandlerSocket)

Visit <http://yoshinorimatsunobu.blogspot.com/2010/10/using-mysql-as-nosql-story-for.html>

Percona server and MariaDB have already integrated HS into their own version.

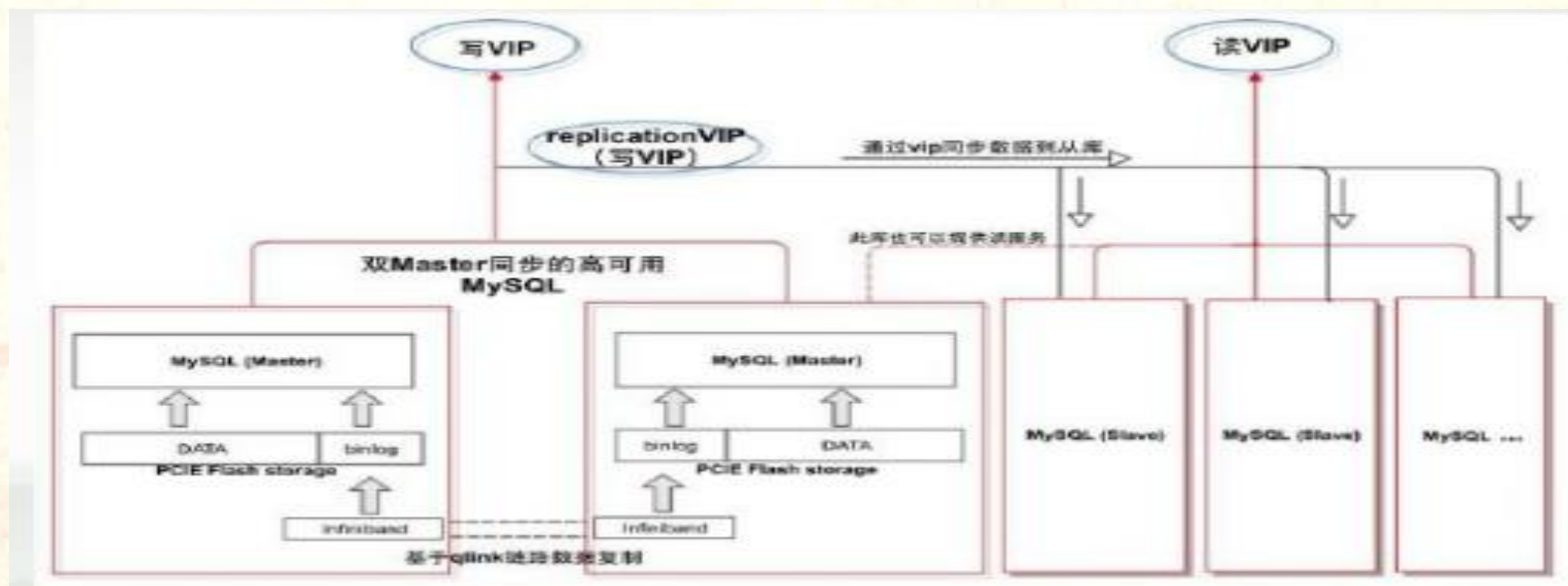
<http://www.mysqlperformanceblog.com/2010/12/14/percona-server-now-both-sql-and-nosql/>

<https://mariadb.com/kb/en/handlersocket/>

Ten: New idea for mysql Integration

As you know in oracle database area HA architect is already mature we can use RAC to do some zero downtime work (like database upgrade or business migration) and for Dataguard technology we can recover standby database in block level (just using block to recover standby database, so with redo logfile and archivelog we can ensure no data loss)

In mysql area we can't use logfile to recover database directly so bin log seems very important for database recover. Recently we communicated with [WOQU](#) and know their new idea for mysql integration:



They put bin log on shared disk to ensure slave can read most recently bin log (avoid some network problem leading lag between two databases) but we still doubt whether InfiniBand network is the best choice in this situation.

To Be Continue . . .