COLLABORATE 12

TECHNOLOGY AND APPLICATIONS FORUM FOR THE ORACLE COMMUNITY

IOUG independent oracle users group

Session: 211

Tuning Linux to Most Efficiently Run Oracle Database on Linux for System z

Time: Wed, Apr 23, 2012

(9:30AM – 10:30 am)

Session Agenda

- Introduction
- Minimize Linux rpms that are installed
- Oracle Resource Consumption Tips
- Monitoring and Tuning Tools

Why Reduce Linux RPMs that are Installed?

- Helps Reduce problems seen by Oracle Support
  - Too many rpms can cause problems
    - Multiple gcc / g++ Versions in Linux MOS Note: **444084.1**
    - May be needed if supporting multiple Oracle Versions
- Helps reduces the amount of Disk space used
- Reduces the Number of Linux services that get created

Minimize the # of rpms Installed

- Review the List of Oracle rpms to Install
- Some Oracle rpms may not be needed such as ODBC Options, or if using Silent Install some of the openmotif rpms.

Install Just The Required rpms That You Need

Required RPMs for a Red Hat 6.1 Install:

binutils-2.20.51.0.2-5.20.el6.s390x.rpm
compat-libcap1-1.10-1.s390.rpm
compat-libstdc++-33-3.2.3-69.el6.s390.rpm
compat-libstdc++-33-3.2.3-69.el6.s390x.rpm
elfutils-libelf-0.152-1.el6.s390x.rpm
elfutils-libelf-devel-0.152-1.el6.s390x.rpm
gcc-4.4.5-6.el6.s390x.rpm
gcc-c++-4.4.5-6.el6.s390x.rpm
glibc-2.12-1.25.el6.s390.rpm
glibc-2.12-1.25.el6.s390x.rpm
glibc-common-2.12-1.25.el6.s390x.rpm
glibc-devel-2.12-1.25.el6.s390.rpm
glibc-devel-2.12-1.25.el6.s390x.rpm
glibc-headers-2.12-1.25.el6.s390x.rpm
ksh-20100621-6.el6.s390x.rpm
libaio-0.3.107-10.el6.s390.rpm
libaio-0.3.107-10.el6.s390x.rpm
libaio-devel-0.3.107-10.el6.s390.rpm
libaio-devel-0.3.107-10.el6.s390x.rpm
libgcc-4.4.5-6.el6.s390.rpm
libgcc-4.4.5-6.el6.s390x.rpm
libstdc++-4.4.5-6.el6.s390.rpm
libstdc++-devel-4.4.5-6.el6.s390x.rpm
make-3.81-19.el6.s390x.rpm
sysstat-9.0.4-18.el6.s390x.rpm

*****If you intend to use the Oracle ODBC driver then you will require the following optional rpms.**

unixODBC-2.2.14-11.el6.s390.rpm
unixODBC-2.2.14-11.el6.s390x.rpm
unixODBC-devel-2.2.14-11.el6.s390.rpm
unixODBC-devel-2.2.14-11.el6.s390x.rpm

***For Linux on System z, you do not need to install the IBM JDK, Oracle supplies the Java for the Database.

Turning Off Unneeded Services – List Services On

```
# chkconfig -l | grep 3:on
cron                    0:off  1:off  2:on   3:on   4:off  5:on   6:off
dbus                    0:off  1:off  2:on   3:on   4:off  5:on   6:off
earlysyslog             0:off  1:off  2:on   3:on   4:off  5:on   6:off
fbset                   0:off  1:on   2:on   3:on   4:off  5:on   6:off
haldaemon               0:off  1:off  2:on   3:on   4:off  5:on   6:off
irq_balancer            0:off  1:on   2:on   3:on   4:off  5:on   6:off
network                 0:off  1:off  2:on   3:on   4:off  5:on   6:off
network-remotefs        0:off  1:off  2:on   3:on   4:off  5:on   6:off
nfs                     0:off  1:off  2:off  3:on   4:off  5:on   6:off
nscd                    0:off  1:off  2:off  3:on   4:off  5:on   6:off
postfix                 0:off  1:off  2:off  3:on   4:off  5:on   6:off
random                  0:off  1:off  2:on   3:on   4:off  5:on   6:off
rpcbind                 0:off  1:off  2:off  3:on   4:off  5:on   6:off
smartd                  0:off  1:off  2:on   3:on   4:off  5:on   6:off
splash                  0:off  1:on   2:on   3:on   4:off  5:on   6:off  S:on
splash_early            0:off  1:off  2:on   3:on   4:off  5:on   6:off
sshd                    0:off  1:off  2:off  3:on   4:off  5:on   6:off
syslog                  0:off  1:off  2:on   3:on   4:off  5:on   6:off
```

Turn off Any Unneeded Services

Keep the golden image as lean as possible in terms of processor usage, some of these services can be turned off with the chkconfig command:

```
# chkconfig fbset off
# chkconfig network-remotefs off
# chkconfig postfix off
# chkconfig splash off
# chkconfig splash_early off
# chkconfig smartd off
```

***** Consider disabling other services like NFS if not used all the time**

Linux rpm considerations

Staying current is extremely important:

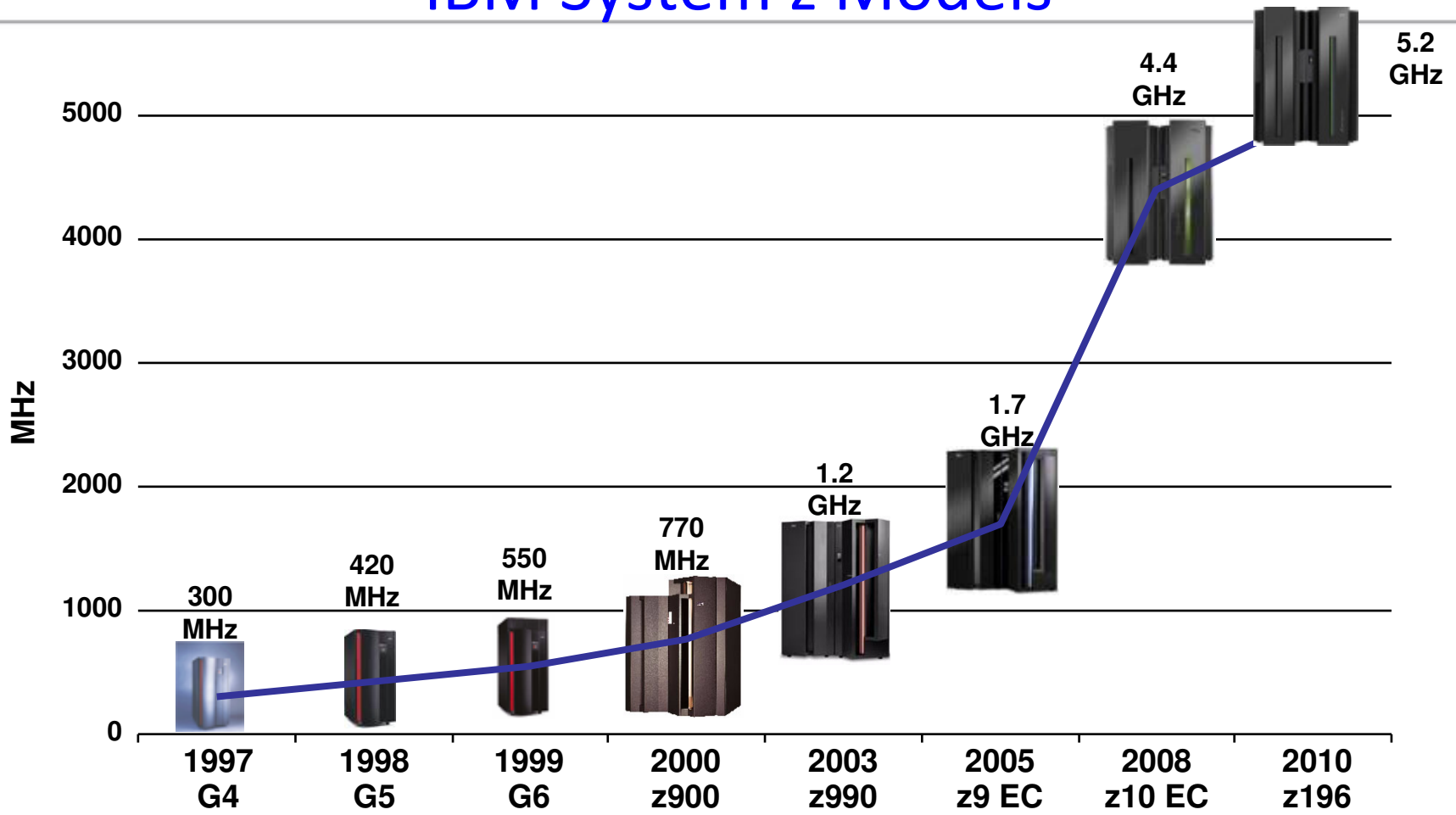
- glibc performance improvements with each release
- gettimeofday() – several vendor improvements

Virtual **D**ynamically-linked **S**hared **O**bject (VDSO) is a shared library provided by the kernel. This allows normal programs to do certain system calls without the usual overhead of system calls like switching address spaces.

On a z196 system for example by using the VDSO implementation **six times** reduction in the function calls are possible.

Newer Linux distributions (RHEL 6, SLES 11) have this feature and it's enabled by default.

IBM System z Models



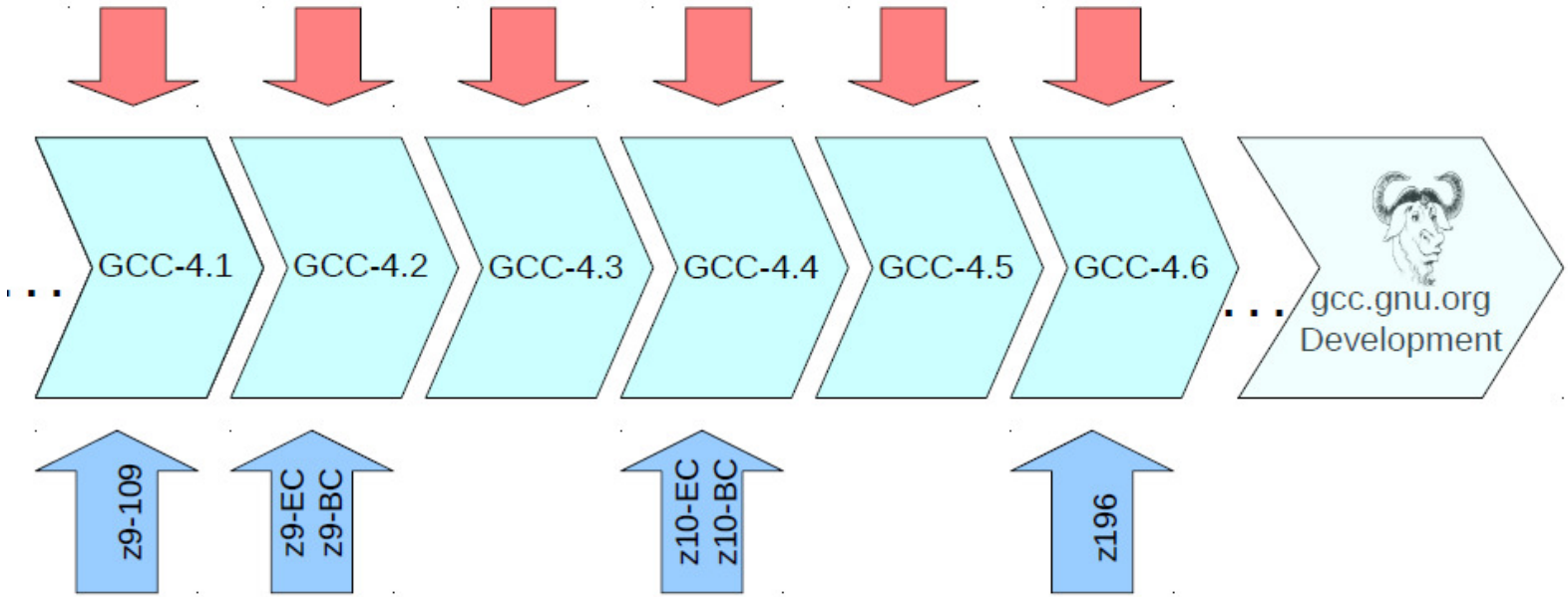
- **G4** – 1st full-custom CMOS S/390®
- **G5** – IEEE-standard BFP; branch target prediction
- **G6** – Copper Technology (Cu BEOL)

- **z900** – Full 64-bit z/Architecture
- **z990** – Superscalar CISC pipeline
- **z9 EC** – System level scaling

- **z10 EC** – Architectural extensions
- **z196** – Additional Architectural extensions and new cache structure

GCC compiler evolution

IBM development provides patches to exploit new GCC features also in Linux on System z (i. e. software DFP, hardware optimized instruction scheduling)



IBM development provides patches to exploit IBM System z and IBM zEnterprise hardware features in new GCC versions (i. e. new instructions, hardware DFP)

GCC versions in Linux on System z supported distributions

GCC version	General available	Included in SUSE distribution	Included in Red Hat distribution
GCC-3.3	05/2003	SLES9 (z990 backport)	
GCC-3.4	04/2004		RHEL4 (z990 support)
GCC-4.0	04/2005		
GCC-4.1	02/2006	SLES10 (z9-109 support)	RHEL5 (z9-109 support)
GCC-4.2	05/2007		
GCC-4.3	05/2008	SLES11 (z10 backport)	
GCC-4.4	04/2009		RHEL6.1 (z196 backport)
GCC-4.5	04/2010	SLES11 SP1 (z196 backport)*	
GCC-4.6	03/2011		
GCC-4.7			

* included in SDK, optional, not supported

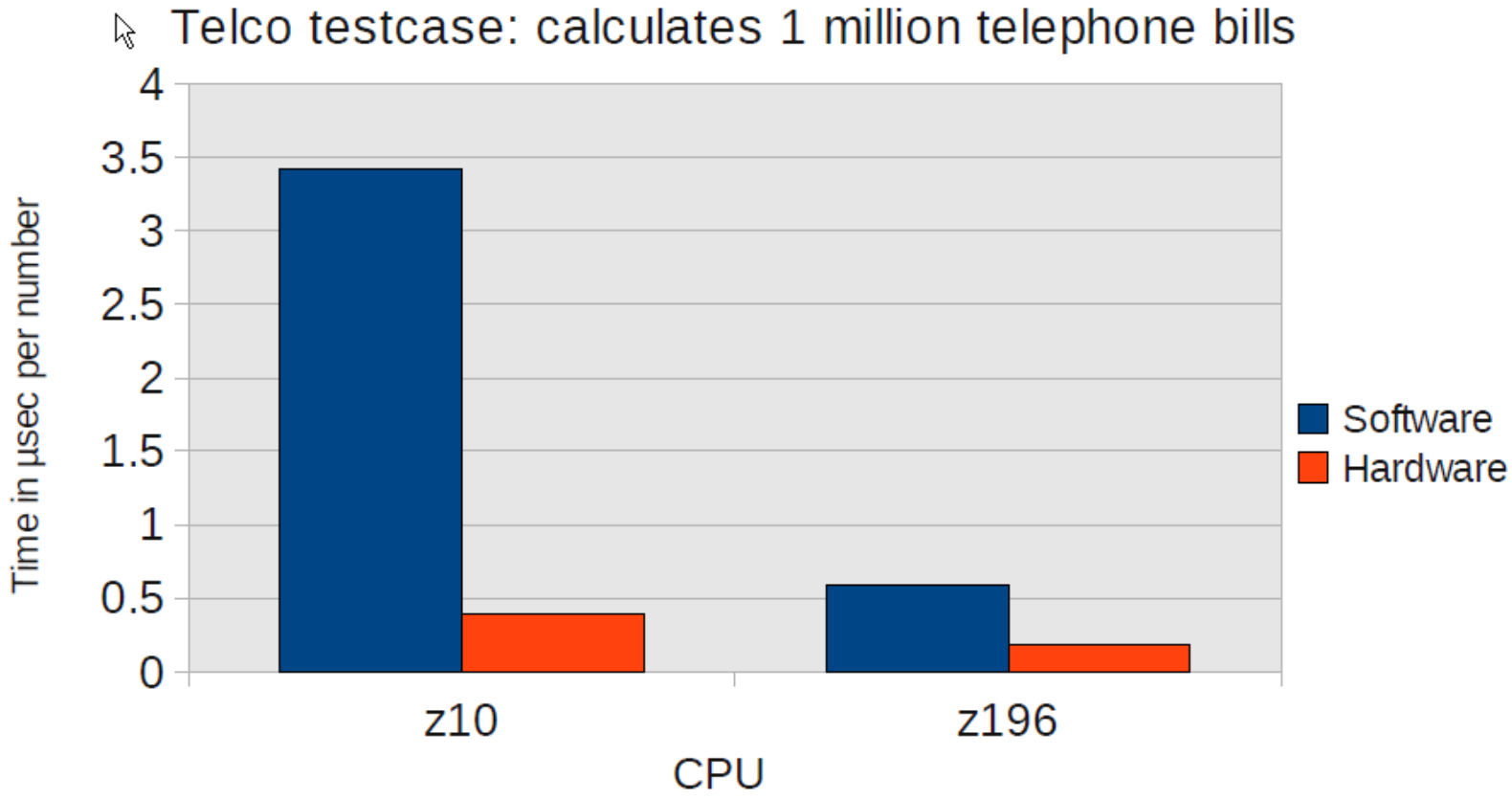
Special GCC compile options for S/390 (31-bit) and System z (64-bit) (2)

- `'-mtune=z900 | z990 | z9-109 | z9-ec | z10 |z196'` generates code optimized for the particular CPU and the set of available instructions.
  - The compiler's instruction scheduling is influenced but not the instruction set.
  - The generated code targeting one CPU type will run on a different mainframe CPU type but may cause a performance degradation there.
  - For eServer zSeries 800 use the 'z900' argument, for eServer zSeries 890 use the 'z990' argument.
- `'-march=z900 | z990 | z9-109 | z9-ec | z10 | z196'` generates code optimized for the particular processor, using the instruction set of the processor.
  - The generated code will run on the target processor type or a later mainframe processor type.
    - So code compiled with 'march=z10' will run on a IBM zEnterprise System but it is not guaranteed that code compiled with 'march=z196' will run on an IBM System z10.

Special GCC compile options for S/390 (31-bit) and System z (64-bit) (3)

- Our experiments show remarkable overall performance improvements.
  - Use '-march', if the generated optimized code is to be executed on only that single target machine type. If more than one target machine is identified, use the argument for the oldest model ('-march' parameter is upward compatible).
  - Use '-mtune' parameter, if the generated optimized code is intended to run optimal on a specific target machine type, but should be runnable on other (previous) machines, too.
  - Defaults coming with the compiler
    - For **RHEL5 GCC-4.1** '-mtune=z9-109' and '-march=z900'.
    - For **SLES10 GCC-4.1** '-mtune=z9-109' and '-march=z900'.
    - For **SLES11 GCC-4.3** the defaults are '-mtune=z9-109' and '-march=z900'.
    - For **RHEL6 GCC-4.4** the defaults are '-mtune=z10' and '-march=z9-109'
    - In other 64-bit environments the defaults are '-mtune=z900' and '-march=z900'.

GCC Compile Options Performance



Advantage of the compiled in hardware implementation can be seen compared with the software solution running on the same system in the following chart.

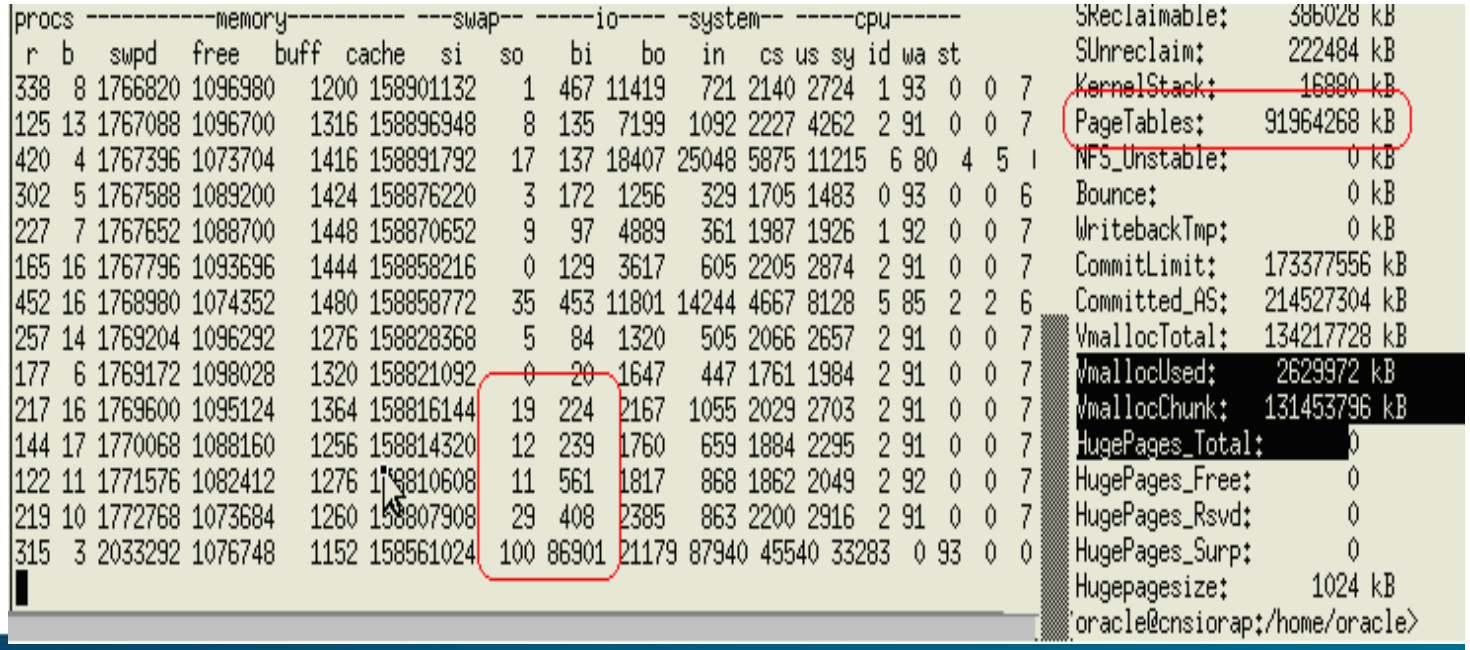
CPU hotplug summary

- This feature improves the performance by
  - sizing the correct amount of processors for a Linux system depending on its current load
  - avoiding the Linux scheduler queue balancing in partial load situations
- Set the minimum and maximum number of CPUs to values which apply to the real workload:
  - Setting `cpu_min` to 2 may be too high
  - `cpu_max` should be set so that it really covers the peaks
- The update interval should be selected carefully
  - a low value will have the effect that reactions to load changes are not immediate
  - a high value has a bad effect if the load does not change very much because each check consumes processor cycles.
- Linux guests under z/VM: use z/VM 5.4
  - Guarantees that stopped processors are no longer included in virtual processor prioritization calculations
  - Ensures share redistribution

Additional CPU Savers $$$

- **Consider Using Linux Huge Pages for Oracle Database Memory**

  →In general 10-15% can be gained by the reduction in CPU usage as well as having a lot more memory for applications that would be consumed in Linux Page Tables…



Networking Performance Considerations (1)

- For Oracle RAC Environments, **choose your MTU size carefully**. Set it to the maximum size supported by all hops on the path to the final destination to avoid fragmentation.
  - Use the **tracepath** command to verify the path MTU size.
  - If the application sends data in chunks of <= 1400 bytes, use an MTU size of 1492:
    - 1400 bytes user data plus protocol overhead.
  - If the application is able to send bigger chunks, use an MTU size of 8992.
  - Sending packets > 1400 with an MTU size of 8992 will increase throughput and save CPU cycles.

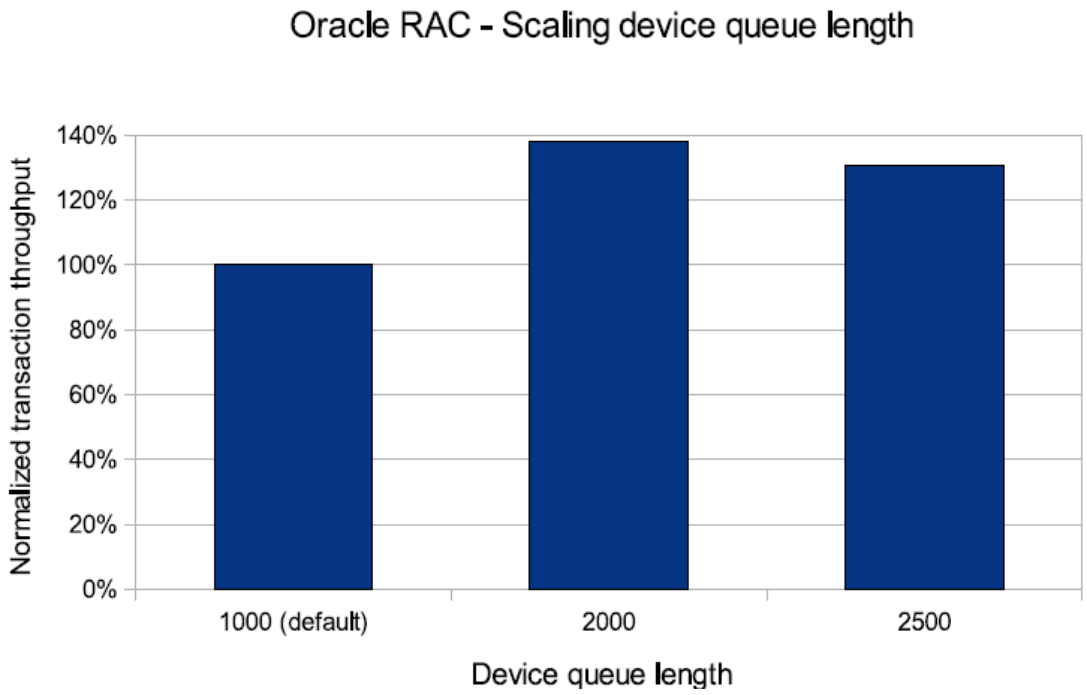
- For HiperSockets, select an MTU size to suit the workload:
  - If the application is capable of sending large packets, a larger MTU size will increase throughput and decrease cpu cycles.
  - An MTU size of 56K is recommended only for data streaming workloads with packets > 32KB.

Networking Performance Considerations (2)

- HiperSockets doesn't require check summing because it is a memory-to-memory operation.
  - The default is sw_checksumming.
  - To save CPU cycles, switch checksumming off for HiperSockets:
    - **Novell SLES10: /etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.F200 add QETH_OPTIONS="checksumming=no_checksumming"**
    - **Novell SLES11: /etc/udev/rules.d/51-qeth-0.0.f200.rules add ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.f200", ATTR{checksumming}="no_checksumming"**
    - **RedHat RHEL4&5: /etc/sysconfig/network-scripts/ifcfg-eth0 add OPTIONS="checksumming=no_checksumming"**

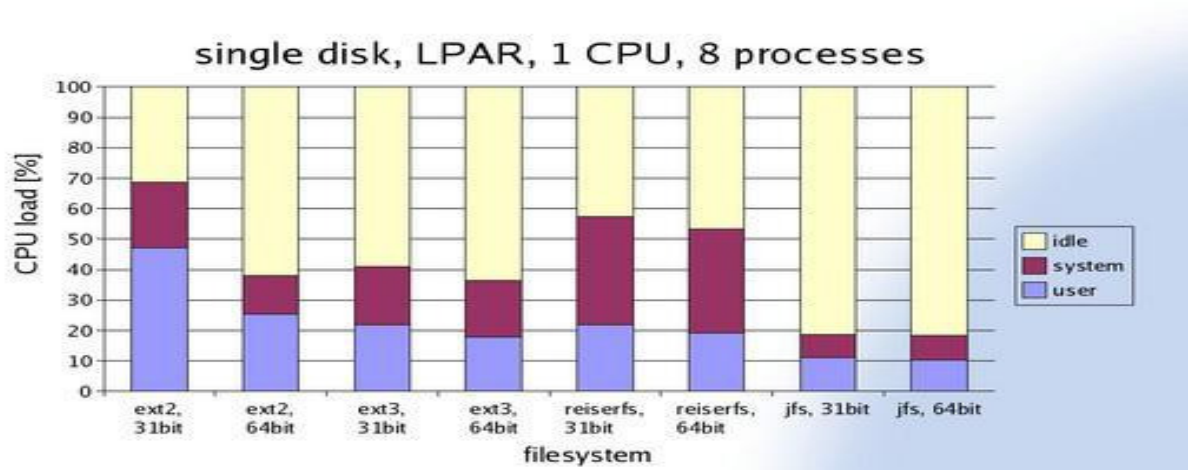
Networking Performance Considerations (3)

- The device queue length should be increased from the default size of 1000 to at least 2000 using sysctl:
  - **sysctl -w net.core.netdev_max_backlog =2000**



File System Options

- EXT2 - most widespread Linux file system.
- EXT3 - evolved from ext2, adds journaling features.
- EXT4 –**only supported with RH 5.6/OL 5.6/SLES 11 or greater** **(recommend to test first)**
- JFS - a port of OS/2 Warp Server jfs to Linux.
- Reiserfs – journaling behavior is comparable to ext3 in order mode.
- **Recommend using ext3 or ext4 due** its journaling capabilities and reduced cpu load
- See latest performance report at: **http://download.boulder.ibm.com/ibmdl/pub/software/dw/linux390/perf/ZSW03027-USEN-00.pdf**



Kernel I/O Scheduler

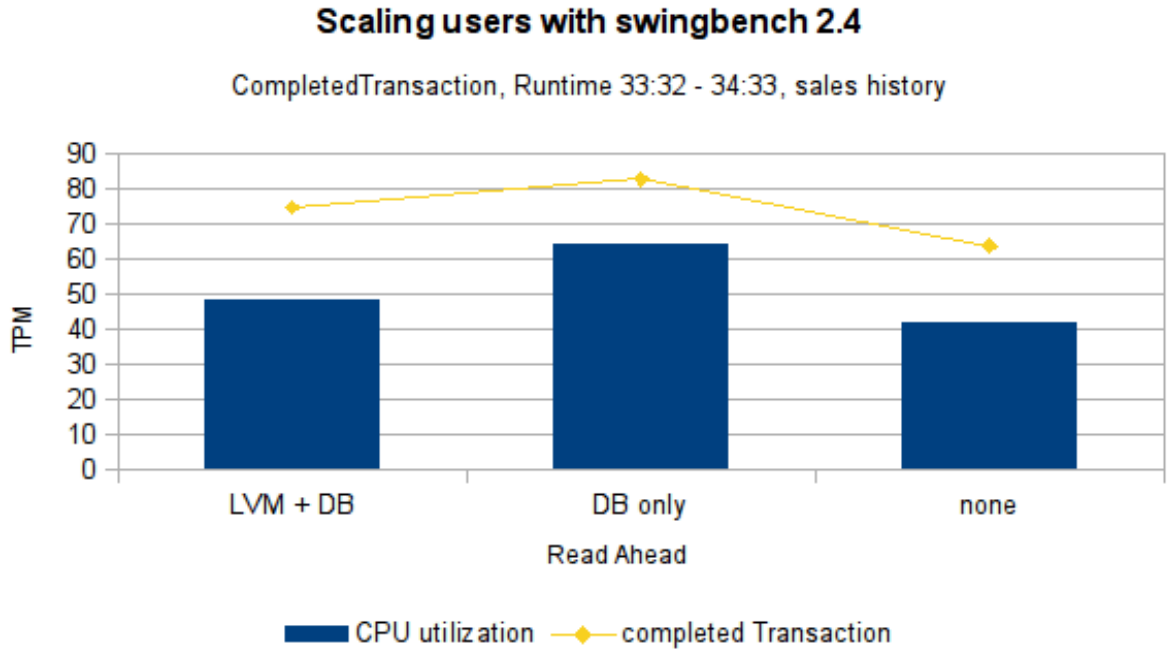
- The I/O scheduler optimizes disk access, the strategy for optimization aims to minimize the number of I/O operations and disk head movements.
- The Linux 2.6 kernel offers a choice of four different I/O schedulers:
  - Noop Scheduler (noop)
  - Deadline Scheduler (deadline)
  - Anticipatory Scheduler (as)
  - Complete Fair Queuing Scheduler (cfq)
- Linux default is the “as” scheduler:
  - Designed to optimize access to physical disks.
  - Not suitable for typical storage servers used in the System z environment
  - Selected by setting the “elevator” boot parameter in /etc/zipl.conf
- Recommended I/O scheduler – **deadline or noop**

Additional CPU Savers $$$

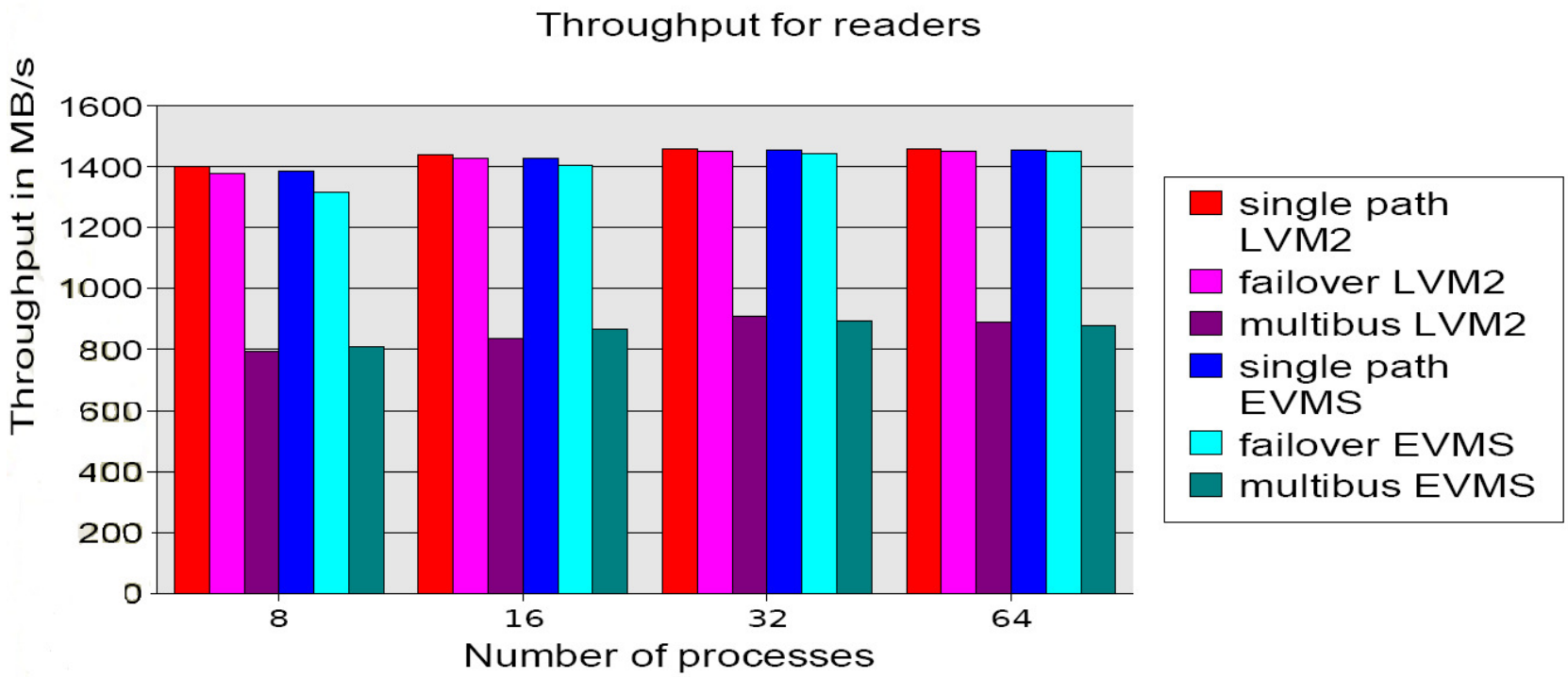
- Investigate the best I/O scheduler for your environment
  - On a Red Hat system we changed I/O scheduler in the **zipl.conf parameters"elevator=noop"** this helped with reducing cpu for the SAN environment we were using.

Additional CPU Savers $$$

- Reduce the Linux Read-Ahead for LVM file systems.
  - **lvchange -r none <lv device name>**



Multipath.conf configuration



SLES 10, Red Hat 5 Multipath with Failover

SLES 11 / Red Hat 6 keep default Multibus

Upgrade to the latest Release 11.2.0.3

- Oracle's **VKTM** process uses slightly less CPU minutes
  - (about **0.08** vs. 0.09 with 11.2.0.2)

- Great improvements with **ora_dia0** process.
  - (about **0.07** sec cpu/minute vs. **0.28** with 11.2.0.2)

- Only Install the database modules that are needed
  - DB installed with **NO** options
    The "gettimeofday" function is called 300 times every 15 seconds.

  - DB installed with **all** options : (java, xml, Text, spatial, APEX, etc .....)
    The "gettimeofday" function is called 1500 times every 15 seconds.

Other Oracle Resource Saving Tips

- RMAN Backups – Consider using Culmative Merged Backups (RMAN Level 1 merged with Level 0) and or Oracle's Advanced Compression.

- Review the v$scheduled_jobs in particular the sql tuning advisor and segment advisor to see the best usage for your environment.

Database cpu Reducing Parmeters

- **filesystemio_options=setall** for databases with file systems to reduce caching to Linux File system cache memory.

- If seeing "resmgr:cpu quantum" wait events may want to disable the resource manager default plan
  - **resource_manager_plan = ''**

"resmgr:cpu quantum Wait Event"

Additionally You need disable the Maintenance Window Resource Plan

```
select window_name,RESOURCE_PLAN
from DBA_SCHEDULER_WINDOWS;
```

WINDOW_NAME	RESOURCE_PLAN
MONDAY_WINDOW	DEFAULT_MAINTENANCE_PLAN

```
execute dbms_scheduler.set_attribute('MONDAY_WINDOW','RESOURCE_PLAN','');
```

WINDOW_NAME	RESOURCE_PLAN
MONDAY_WINDOW	

Investigate Locking Table Statistics for Large Tables

```
DBMS_STATS.UNLOCK_TABLE_STATS(ownname => 'USERS', tabname => 'XXX');

DBMS_STATS.GATHER_TABLE_STATS(ownname => 'USERS ', tabname => ' XXX',
    estimate_percent=>1, cascade =>TRUE, degree =>4);

DBMS_STATS.LOCK_TABLE_STATS(ownname => 'USERS', tabname => 'XXX');
```

Questions