# Advocacy for the OpenSource Relational Data Base Management System Over FreeBSD

Jorge A. Medina - Santiago/Chile

Computer Science Research Crew
http://www.bsdchile.cl ®

Feb. 20 , 2009

# Index

# Way FreeBSD?

1. The BSD license it's realy "Free" this it's important.
2. The OS is released under BSD license
   other important products like PostgreSQL too.
3. Installing easy and fast.
4. Upgrading easy bandwidth-depended but realy fast
   (freebsd-update,csup,portupgrade).
5. Software necesary ports tree (portsanp fetch,extract,update).
6. Easy Kernel build (make buildkernel, make installkernel).
7. Easy User Land build (make buildworld, make installworld).
8. Basic Virtualization or chrooted environment
   more secure with FreeBSD Jails.
9. Basically the core it's realy realy more powerfull than others OS.
   Branch-7 Re-designed the FreeBSD kernel
   as a multi-threaded system for "next generation" SMP support
10. And many many others.

# Index

# How it's possible[1]?

- FreeBSD 7.0 was released, and the first major release in 2 years.
- FreeBSD 7.1 brings major changes to the BSD and open source operating system landscape.
- FreeBSD 8.0 comming soon, it's the current available for testing

*The begin*
*"Last week, approximately 20 BSD developers got together and discussed how to move FreeBSD's SMP support to the next level. Our effort will be largely based on the work that has been done in BSD/OS, which should make things go much more smoothly than they otherwise might, but we still expect -current to be destabilized for an extended period of time."*
*(Jason Evans, email to freebsd-current list, 19 June 2000.)*

[1] Source: Kris Kennaway FreeBSD Project committer since 1999.

# The Result

- **Goal:** Re-design the FreeBSD kernel as a multi-threaded system, for "next generation" SMP support (June 2000).
- Multiple CPUs must be able to execute kernel code in parallel
- Balance the performance needs of Uni-Processor (UP) and SMP systems (not always different needs)
- A major challenge...
- ...now complete

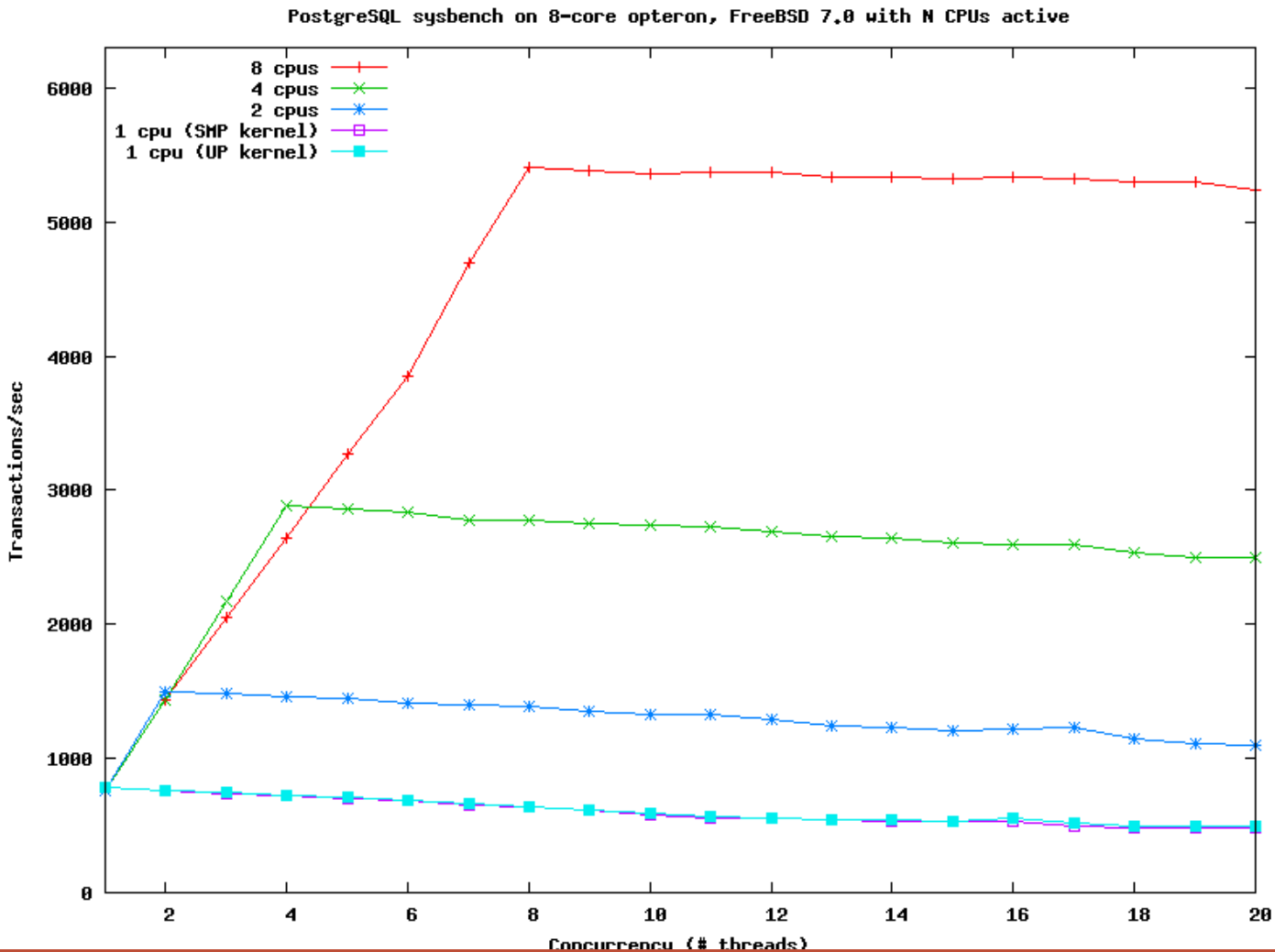
# SQL DataBase Performance

- "Online transaction processing" benchmark; /usr/ports/benchmarks/sysbench
- Transaction-based queries
- Read-only: no disk access to avoid benchmarking disk performance
- Clients and servers on the same system
- PostgreSQL 8.2.4 (process-based + System 5 Inter-Process Communication (IPC)
- MySQL 5.0.45 (thread-based)
- Test hardware:
  - 2-core Opteron (amd64 mode) 2.2GHz CPUs, 4 GB RAM
  - 4-core Xeon E5320 (i386 mode) 1.8GHz CPUs, 3.5GB RAM

# Performance Of PostgreSQL

- The ULE scheduler has significantly better performance than old 4BSD (historical BSD scheduler)
  - Better interactivity for desktop users also
  - 4BSD will remain the default in 7.0, changing in 7.1
  - You can easily switch to ULE by recompiling your kernel
- PostgreSQL with ULE has linear scaling to 8 CPUs and minimal degradation at higher loads; close to ideal performance from the hardware.
- No significant performance problems in the FreeBSD 7 kernel on this workload

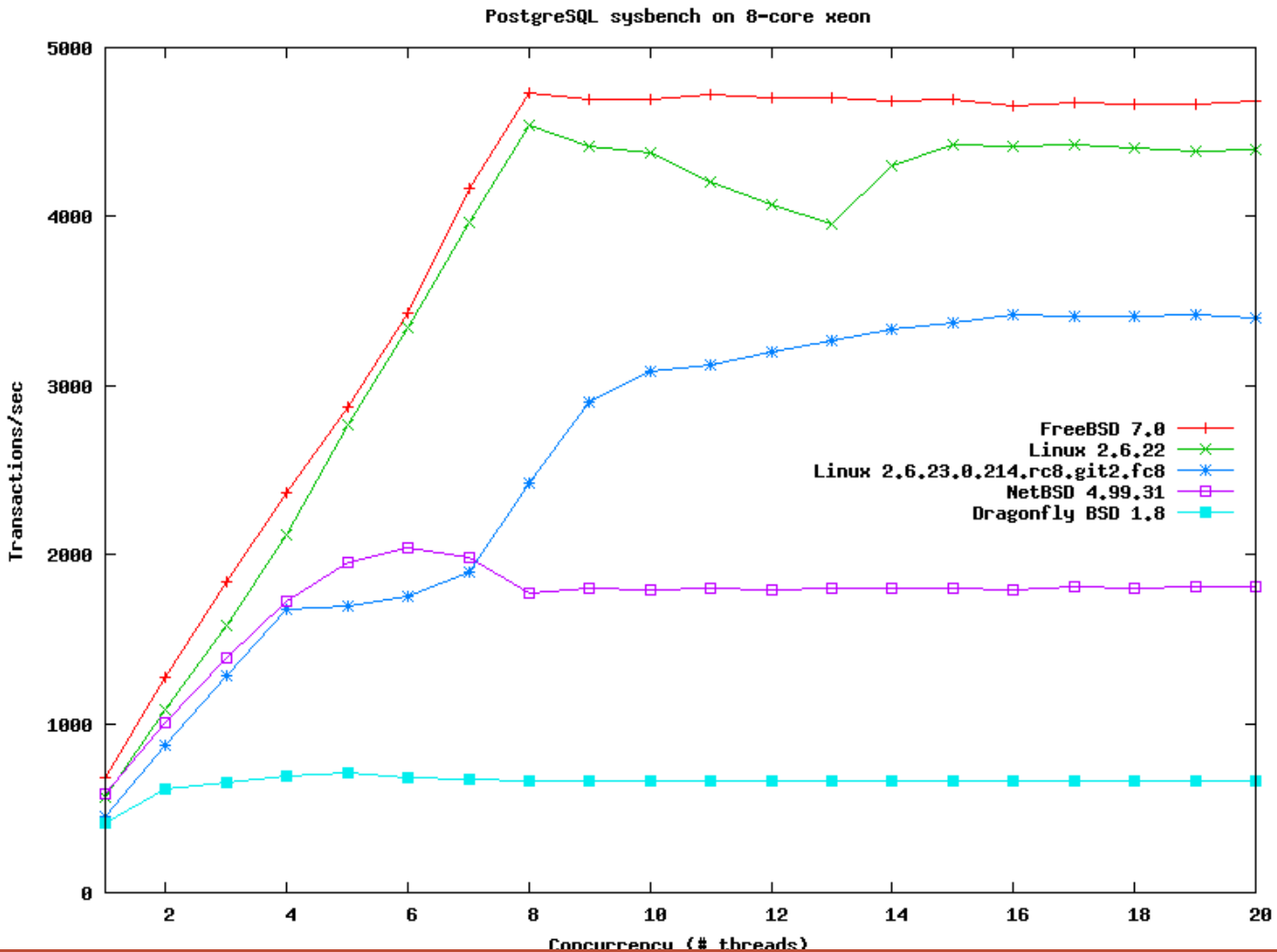
# Scaling with varying number of CPUs



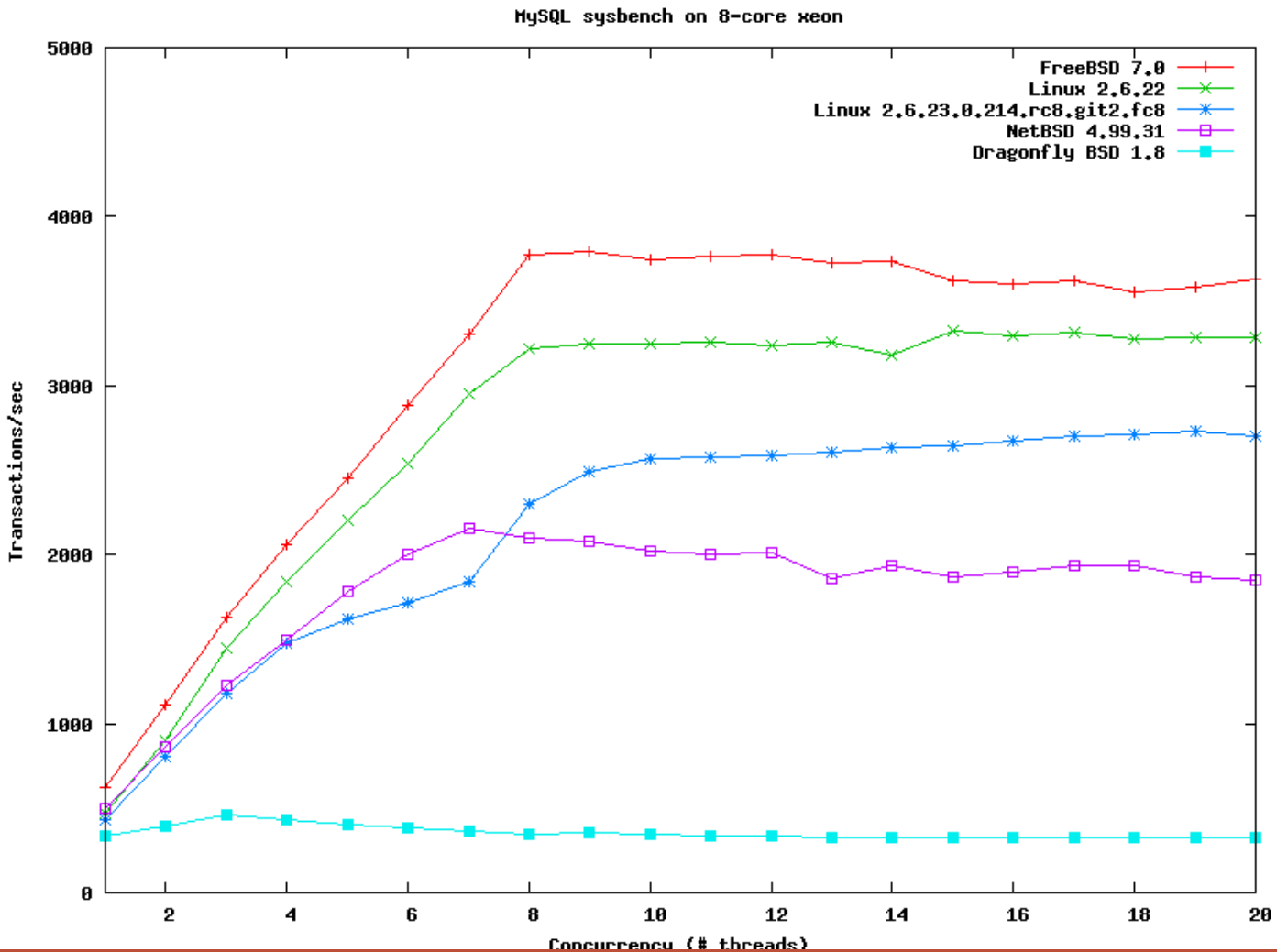
## **Note:** Scaling with varying number of CPUs

- Performance from 1 2 4 8 CPUs scales linearly
- Consistently stable performance at high loads
- No significant overhead from SMP kernel on UP system

# FreeBSD vs other operating systems: PostgreSQL



# FreeBSD vs other operating systems: MySQL



## MySQL performance

- Again, linear scaling up to 8 client threads (= # CPUs)
- The degradation above 8 threads is due to scaling problems within MySQL (not a FreeBSD kernel issue)
- Heavy contention on pthread mutexes within the application
- Recent change to libpthread to reduce the performance loss from heavily contended pthread mutexes
  - Non-portable "adaptive" mutex type defined by glibc, used by MySQL
- Ultimately a MySQL architectural problem
- On this benchmark PostgreSQL is 35% - 45% faster than MySQL at all loads

# Index

# TCP Tuning

Add these to **/etc/sysctl.conf**

```
kern.ipc.maxsockbuf=16777216
net.inet.tcp.rfc1323=1
```

FreeBSD 7.x added TCP send and receive buffer autotuning.
There are some additional settings to modify.
(The default for these is 256 KB, which is too small):

```
net.inet.tcp.sendbuf_max=16777216
net.inet.tcp.recvbuf_max=16777216
```

Here are some other new settings in 7.x to know about. Defaults for these should be fine.

```
net.inet.tcp.sendbuf_auto=1        # Send buffer autotuning enabled by default
net.inet.tcp.sendbuf_inc=8192      # step size
net.inet.tcp.recvbuf_auto=1        # enabled
net.inet.tcp.recvbuf_inc=16384     # step size
```

# TCP Tuning

FreeBSD's TCP has something called inflight limiting turned on by default.
This is good for modem connections, but can be detrimental to TCP throughput in some high-speed situations. If you want "normal" TCP Reno-like behavior, set this:

```
net.inet.tcp.inflight.enable=0
```

By default, FreeBSD caches connection details such as the slow start threshold and the congestion windows size from the previous connection to the same host for 1 hour. While this is a good idea for a web server, it makes it hard to do network throughput testing, as 1 large congestion event will throttle performance for the next hour.
To reduce this effect, set this:

```
net.inet.tcp.hostcache.expire=1
```

This will still cache values for 5 minutes.

# Important Options for DataBase Host

To have these settings persist over reboots, modify /etc/sysctl.conf.
The remaining semaphore settings are read-only as far as sysctl is concerned, but can be changed before boot using the loader prompt:

```
(loader) set kern.ipc.semmni=256
(loader) set kern.ipc.semmns=512
(loader) set kern.ipc.semmnu=256
```

Similarly these can be saved between reboots in /boot/loader.conf.
You might also want to configure your kernel to lock shared memory into RAM and prevent it from being paged out to swap.
This can be accomplished using the sysctl setting

```
kern.ipc.shm_use_phys
```

# Important Options for DataBase Host

If running in FreeBSD jails by enabling sysctl's

security.jail.sysvipc_allowed

Postmasters running in different jails should be run by different operating system users. This improves security because it prevents non-root users from interfering with shared memory or semaphores in a different jail, and it allows the PostgreSQL IPC cleanup code to function properly. (In FreeBSD 6.0 and later the IPC cleanup code doesn't properly detect processes in other jails, preventing the running of postmasters on the same port in different jails.)

Others System Controls need tuning can be:

kern.ipc.shmall=32768
kern.ipc.shmmax=134217728
kern.ipc.semmap=256

**See:** Table 5.1

# Index

# MySQL

MySQL dont have a complicated configuration with difficult decisions.
But its important use InnoDB, instead MyIsam engine this can be set at CREATE command.
It will set by default my.cf:

```
default_table_type = InnoDB
```

**Note:** Recent changes in libpthread, reduce the performance because
can't portable "adaptive" mutex in glib pgsql IPC is 35% more faster, only over FreeBSD.

# Index

# PostgreSQL

PostgreSQL ("pgsql" ahead) pgsql will require some manipulate parameters in some files after install like:

(.cshrc)

```
setenv PGDATA /usr/local/pgsql/data
```

**Now** pg_ctl command work without -D flag

(postgresql.conf)

```
max_connections = 100
shared_buffers = 32MB                   # min 128kB or max_connections*16kB
                                        # (change requires restart)
temp_buffers = 8MB                      # min 800kB
                                        # Note: Increasing max_prepared_transactions costs ≈600 bytes of shared memory
                                        # per transaction slot, plus lock space (see max_locks_per_transaction).
work_mem = 1MB                          # min 64kB
maintenance_work_mem = 16MB             # min 1MB
max_stack_depth = 2MB                   # min 100kB
max_fsm_pages = 204800                  # min max_fsm_relations*16, 6 bytes each
```

(pg_hba.conf)

**Note:** PostgreSQL don't used a posix thread to attend each of the incoming connections instead it use the complete instance of DataBase, this require specific tuning depending of system hardware (RAM) in this case.

# PostgreSQL

**Table 5.1:** System V IPC parameters

| Name | Description | Reasonable values |
|---|---|---|
| SHMMAX | Maximum size of shared memory segment (bytes) | at least several megabytes (see text) |
| SHMMIN | Minimum size of shared memory segment (bytes) | 1 |
| SHMALL | Total amount of shared memory available (bytes or pages) | if bytes, same as SHMMAX; if pages, ceil(SHMMAX/PAGE_SIZE) |
| SHMSEG | Maximum number of shared memory segments per process | only 1 segment is needed, but the default is much higher |
| SHMMNI | Maximum number of shared memory segments system-wide | like SHMSEG plus room for other applications |
| SEMMNI | Maximum number of semaphore identifiers (i.e., sets) | at least ceil(max_connections / 16) |
| SEMMNS | Maximum number of semaphores system-wide | ceil(max_connections / 16)  17 plus room for other applications |
| SEMMSL | Maximum number of semaphores per set | at least 17 |
| SEMMAP | Number of entries in semaphore map | In some cases it might also be necessary to increase in order to SMMNS (SMMNS/2) |
| SEMVMX | Maximum value of semaphore | at least 1000 (The default is often 32767, don't change unless forced to) |

# Resource Limits

The file **/etc/login.conf** controls the various resource limits set during login.
See the operating system documentation for details. The relevant parameters are maxproc, openfiles, and datasize. For example:

```
default:\
...
                :datasize-cur=256M:\
                :maxproc-cur=256:\
                :openfiles-cur=256:\
...
```

(-cur is the soft limit. Append -max to set the hard limit.)
Kernels can also have system-wide limits on some resources.

# Index

## Summary

- FreeBSD 7.x and 8 brings FreeBSD back to the forefront of OS performance on modern hardware (it's good to be back).
- Provides advanced features not available in other open source operating systems.
- An attractive platform for high hardware.
- Better performance of RDBMS OpenSource like PostgreSQL and MySQL.
- Disponibility of many applications ports, from source(ports tree) and binary(packages).
- Good software management.
- and ZFS its coming :)

## More Information

- http://www.freebsd.org/
- http://people.freebsd.org/~kris/
- http://www.postgresql.org/
- http://www.bsdchile.cl/

# THE END

Thanks for coming