(Ab)use the Kernel: What a database server can do to your kernel

Stewart Smith
Core Drizzle Developer
Sun Microsystems Inc

Kernel Conference
Australia 2009
Brisbane, 15-17 July

(Ab)use the Kernel: What a database server can do to your kernel

Stewart Smith
Dark Lord of the Drizzle Kernel
Sun Microsystems Inc

Kernel Conference
Australia 2009
Brisbane, 15-17 July

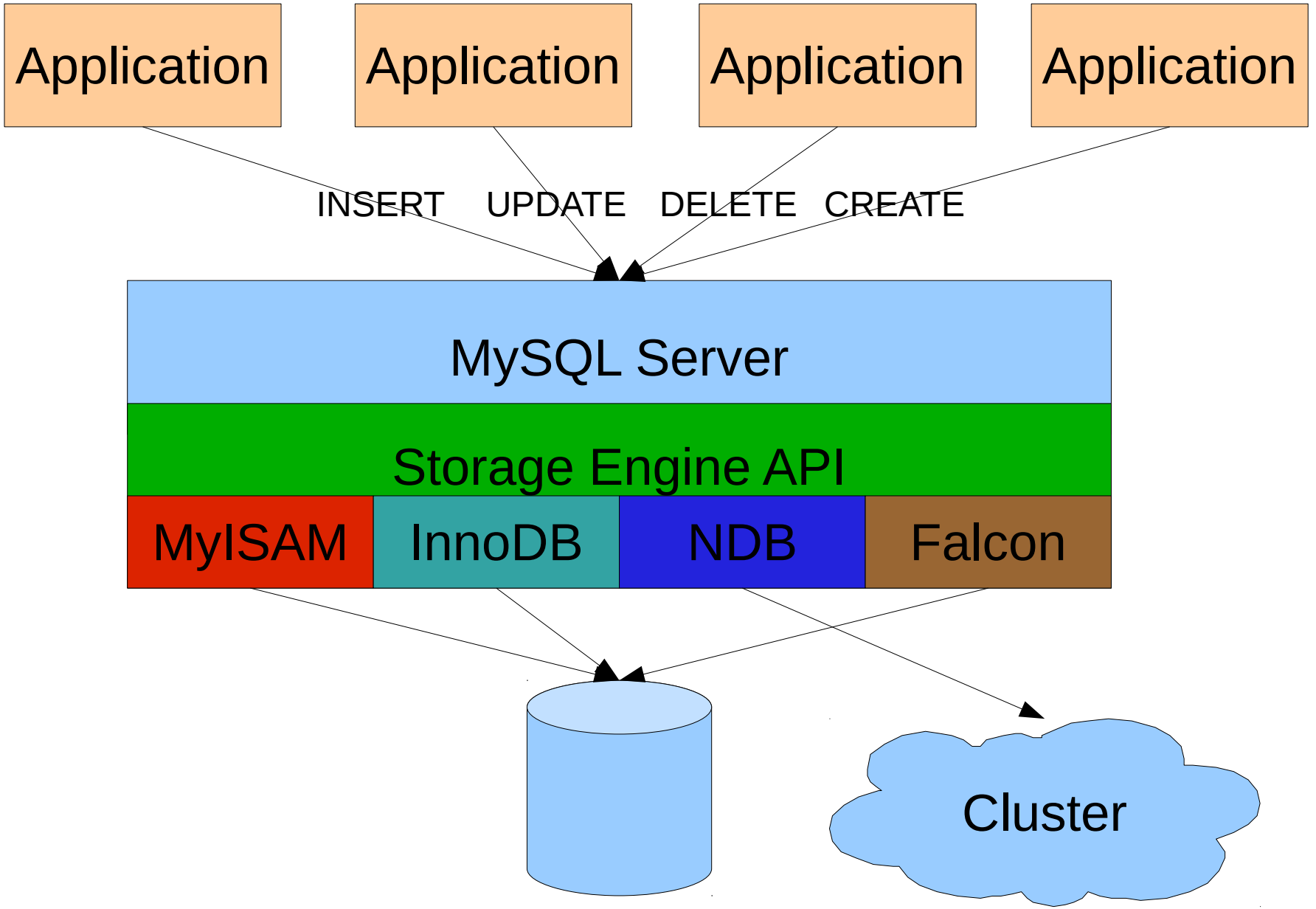
Databases

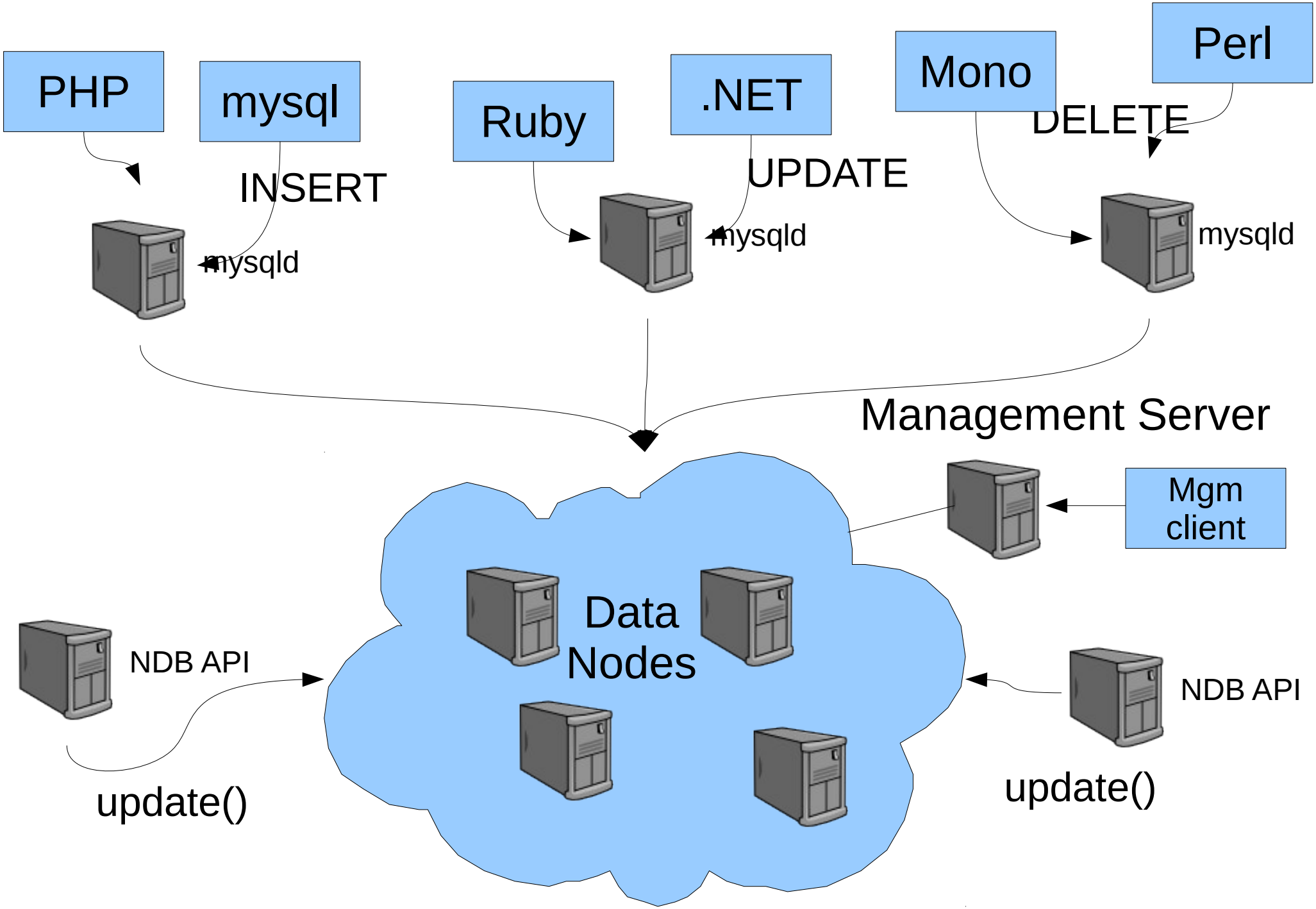
- MySQL
  - InnoDB
- MySQL Cluster (NDB)
  - HA Clustered Database
- Drizzle

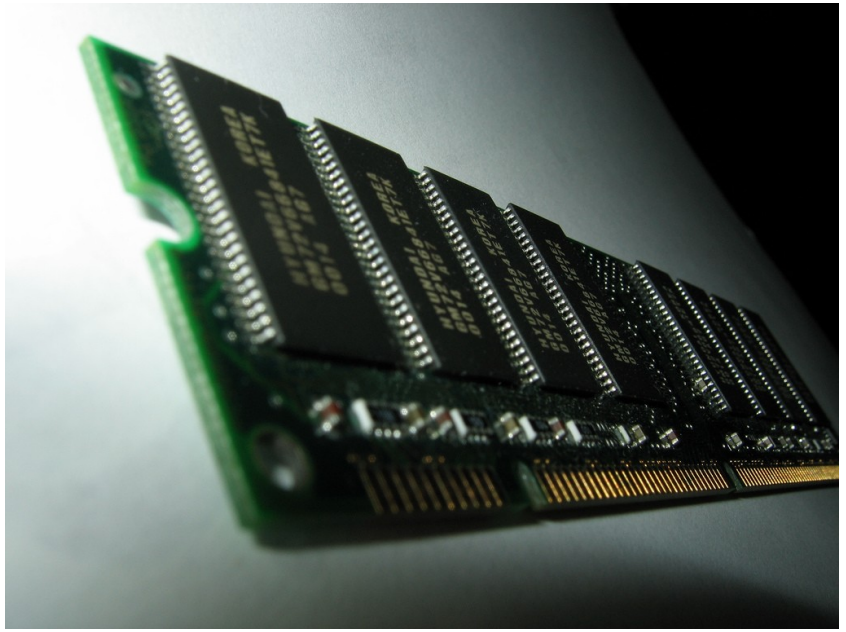
What is MySQL

- This small database that is used by a couple of people

What is MySQL Cluster?


Application
Application
Application
Application
INSERT
UPDATE
DELETE
CREATE
MySQL Server
Storage Engine API
MyISAM
InnoDB
NDB
Falcon
Cluster

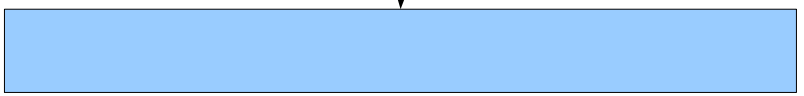

PHP
mysql
Ruby
.NET
Mono
Perl
INSERT
UPDATE
DELETE
mysqld
mysqld
mysqld
Management Server
Mgm client
Data Nodes
NDB API
NDB API
update()
update()



Row

in memory part

on disk part



Checkpointing

- Checkpoint to disk
- On Cluster failure,
  - Recover to previous Global CheckPoint (GCP)
- GCP is a Local Check Point (LCP) + REDO
- Disk data is Data + UNDO
- Durability is 2PC across replicas
  - Txn committed when in memory on all replicas

Memory allocation

- Since in memory data, malloc() large amounts
- Deterministic means not swapping
- Lock pgaes
- 32GB machine
  - 31.5GB locked

Disk IO

- IO with that much locked memory
- O_DIRECT
- Fsync() latency
  - consistency

Network IO

- Sometimes...
  - 1 CPU for network interrupts
  - Spinning better than waiting for IO
  - Must run DB on different CPUs than network interrupts
- Bind specific threads to specific CPUs

Drizzle Goals

- Pluggable
- Infrastructure Aware
- Multi core/concurrency
- Focus on Web Applications
  - Enable others
- Modernise codebase for Managability
  - C++, STL, reuse libraries
- Infrastructure database

Sun's Team Values

- Have open and well documented interfaces
- Have transparent goals and processes, that are communicated publicly
- Have fun and encourage collaboration
- Remove barriers to contribution and participation for anyone
- Enable contributors to build a business around Drizzle

Target OSs

- Linux x86-64, x86 also PowerPC, SPARC
- Solaris
  - OpenSolaris (x86 and SPARC)
  - Solaris 10
- MacOS X
  - Don't care for production, just devs

- Wishlist: OpenBSD, FreeBSD

What does a web site do to a database?

- 1000 simultaneous connections not uncommon
- 10,000 also exists
- TCP versus UDS
- select(), poll(), epoll()
  - Everybody having their own epoll() is not helpful
  - Or different behaviour
- Pluggable schedulers
  - Thread-per-connection
  - Pool-of-threads

Length of connections

- Good web pages <5 SQL queries

Improving Scheduler

- Pool of threads switches execution on blocking network IO
  - Blocking disk io....
  - Moving from thread-per-connection model to something else
- getcontext(), setcontext()

InnoDB and IO

- O_DIRECT from buffer pool
- Data files, log files
- Sometimes file-per-table
- Preallocation
- fsync()

```
#ifdef HAVE_DARWIN_THREADS
# ifdef F_FULLFSYNC
   /* This executable has been compiled on Mac OS X 10.3 or later.
   Assume that F_FULLFSYNC is available at run-time. */
   srv_have_fullfsync = TRUE;
# else /* F_FULLFSYNC */
   /* This executable has been compiled on Mac OS X 10.2
   or earlier.  Determine if the executable is running
   on Mac OS X 10.3 or later. */
   struct utsname utsname;
   if (uname(&utsname)) {
      fputs("InnoDB: cannot determine Mac OS X version!\n", stderr);
   } else {
      srv_have_fullfsync = strcmp(utsname.release, "7.") >= 0;
   }
   if (!srv_have_fullfsync) {
      fputs("InnoDB: On Mac OS X, fsync() may be"
            " broken on internal drives,\n"
            "InnoDB: making transactions unsafe!\n", stderr);
   }
# endif /* F_FULLFSYNC */
#endif /* HAVE_DARWIN_THREADS */
```

```
#if defined(HAVE_DARWIN_THREADS)
# ifndef F_FULLFSYNC
        /* The following definition is from the Mac OS X 10.3 <sys/fcntl.h> */
#  define F_FULLFSYNC 51 /* fsync + ask the drive to flush to the media */
# elif F_FULLFSYNC != 51
#  error "F_FULLFSYNC != 51: ABI incompatibility with Mac OS X 10.3"
# endif
        /* Apple has disabled fsync() for internal disk drives in OS X. That
        caused corruption for a user when he tested a power outage. Let us in
        OS X use a nonstandard flush method recommended by an Apple
        engineer. */

        if (!srv_have_fullfsync) {
                /* If we are not on an operating system that supports this,
                then fall back to a plain fsync. */

                ret = fsync(file);
        } else {
                ret = fcntl(file, F_FULLFSYNC, NULL);

                if (ret) {
                        /* If we are not on a file system that supports this,
                        then fall back to a plain fsync. */
                        ret = fsync(file);
                }
        }
#elif HAVE_FDATASYNC
        ret = fdatasync(file);
#else
        /*      fprintf(stderr, "Flushing to file %p\n", file); */
        ret = fsync(file);
#endif
```

Replication (MySQL)

- Linear (buffered) writes
  - Constantly extending file
- Limit on file size
- Deletes old files

Replication (Drizzle)

- We have the opportunity to get it right
  - i.e. avoiding the file system as much as possible

Virtualization

Virtualization

- Is a fad

Virtualization

- Is a fad
- Single digit perf drops across 1000s of machines....

Future Directions

- MySQL/Drizzle
  - SSD
    - TRIM
  - Higher CPU counts
  - More parallel IO
  - Many low cost machines, not few big ones
- NDB (MySQL Cluster)
  - Ever increasing main memory
  - Mix of machine sizes
  - More cores

Summary

- Different interfaces across OSs is annoying
  - Especially if you're not Linux
- We don't trust VM: O_DIRECT it is
- Lots of pages locked (buffer pool)
-
- fsync() must work
- Many TCP connections (think 1,000-10,000)
  - Which can be many threads

Drizzle

- http://drizzle.org/
- #drizzle on FreeNode
- Automated performance graphs
  - http://drizzle.org/performance/
- Buildbot
- Hudson

Q&A