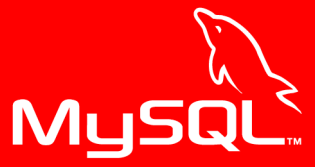

ORACLE®



ORACLE®

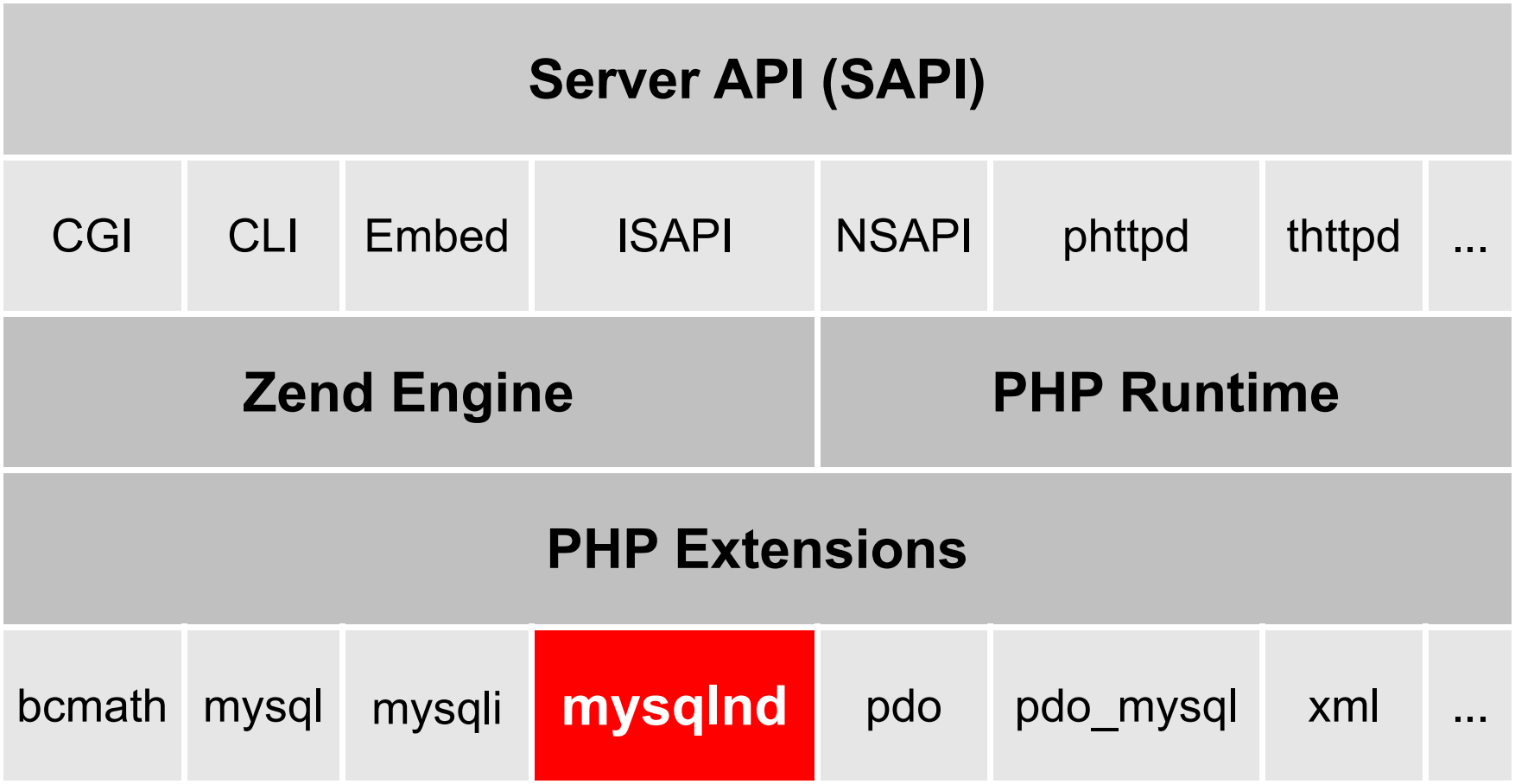
MySQL Client Side Caching

Johannes Schlüter
Twitter: @phperror
MySQL Engineering – Connectors and Client Connectivity

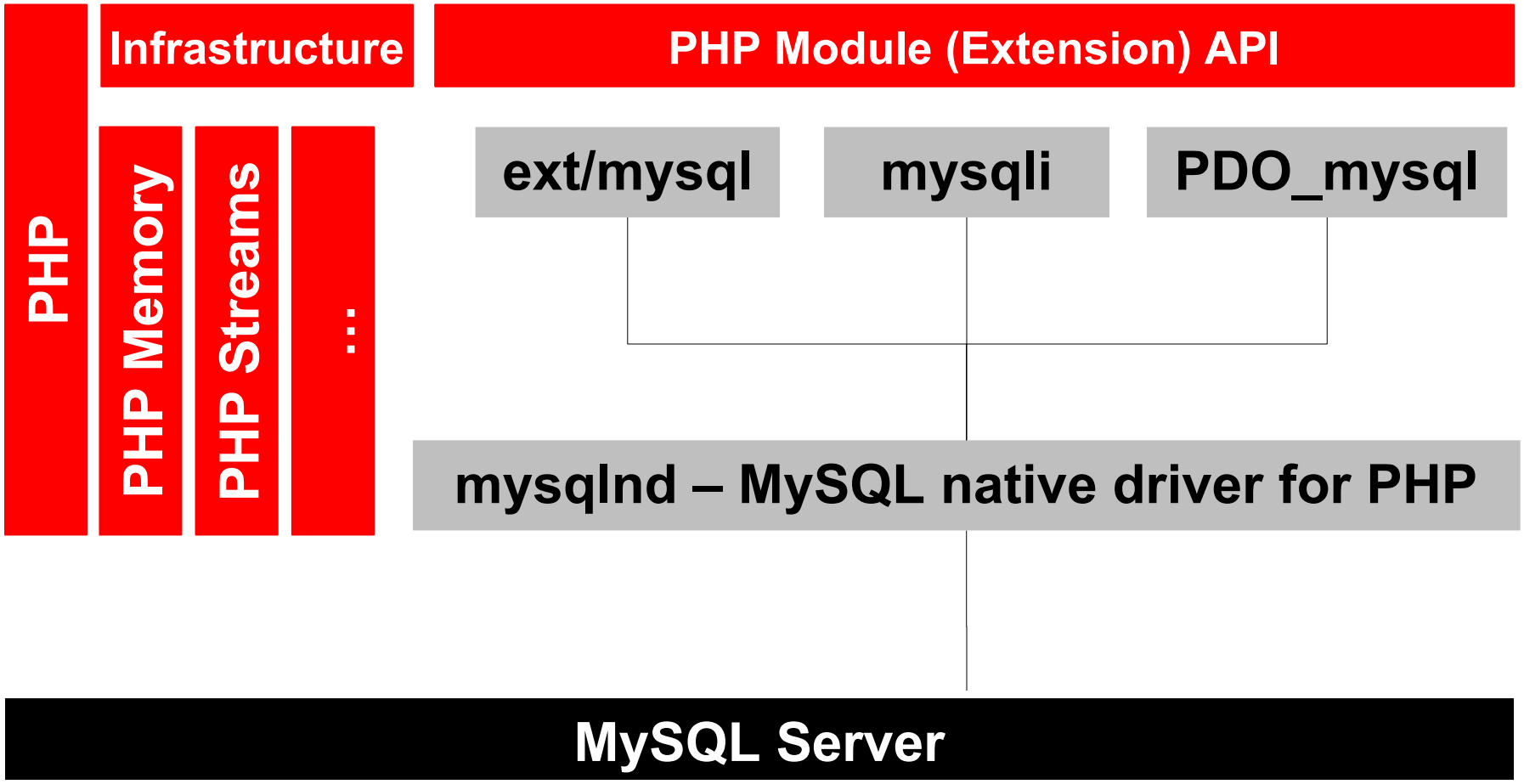
```
# pecl install mysqlnd_qc-beta
```

Gracias por vuestra atención!

mysqlnd


Server API (SAPI)
CGI
CLI
Embed
ISAPI
NSAPI
phttpd
thttpd
...
Zend Engine
PHP Runtime
PHP Extensions
bcmath
mysql
mysqli
mysqlnd
pdo
pdo_mysql
xml
...

PHP 5.3 and mysqlnd



Building PHP with mysqlnd

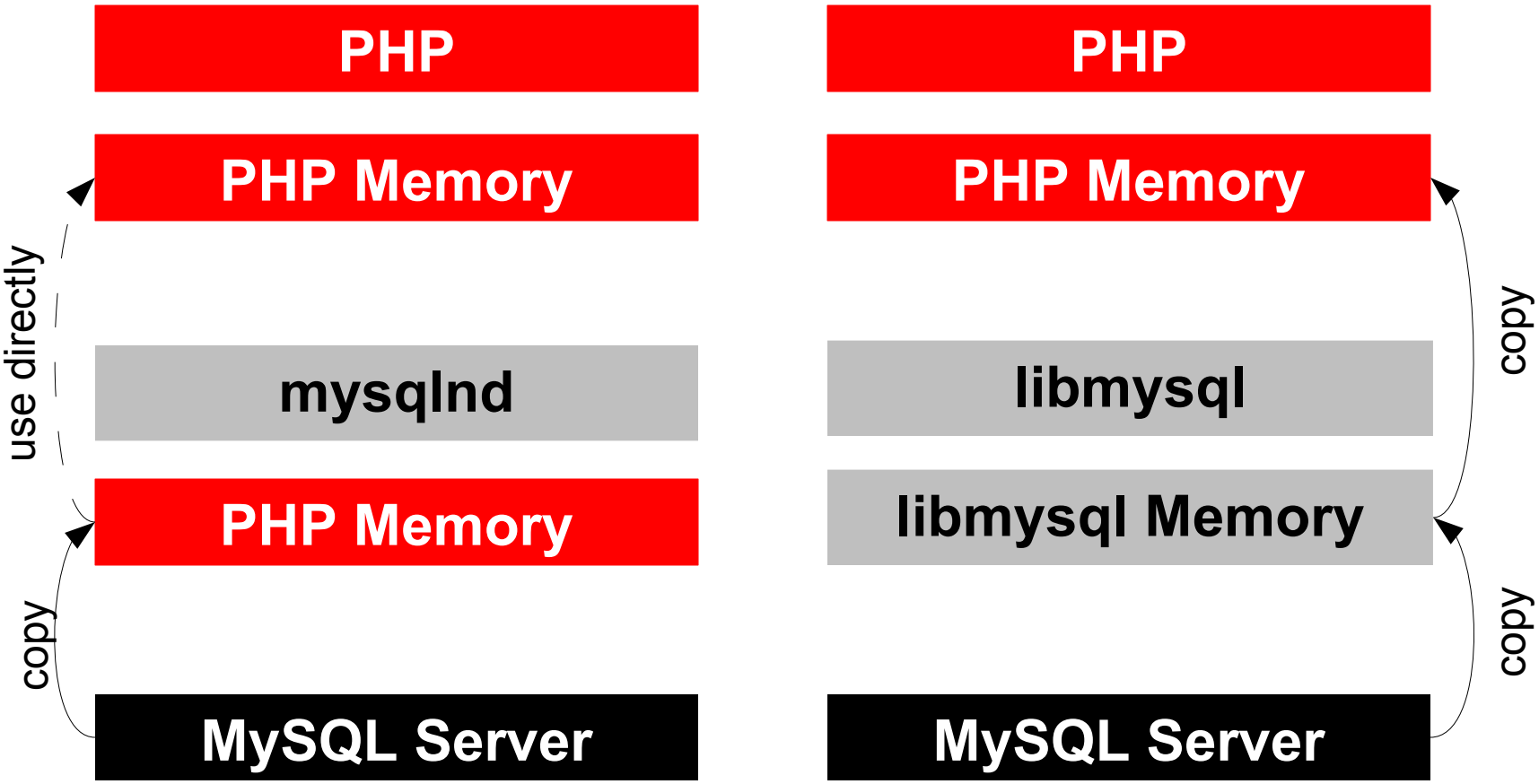
- ./configure \
  --with-mysql=mysqlnd \
  --with-mysqli=msqlnd \
  --with-pdo-mysql=mysqlnd

- Default on Windows and some distributions

mysqli

Mysqli Support	enabled
Client API library version	mysqlnd 5.0.7-dev - 091210 - $Revision: 296270 $
Active Persistent Links	0
Inactive Persistent Links	0
Active Links	9

mysqlnd vs. libmysql



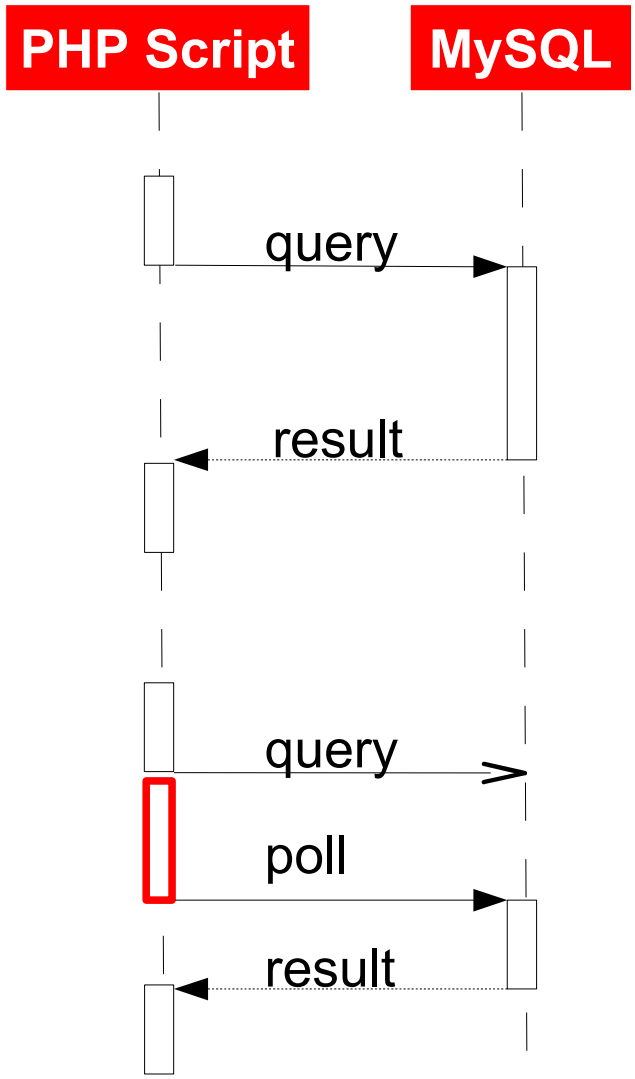
mysqlnd Statistics

Client statistics	
bytes_sent	5381679
bytes_received	39375881
packets_sent	16117
packets_received	469633
protocol_overhead_in	1878532
protocol_overhead_out	64468
bytes_received_ok_packet	18436
bytes_received_eof_packet	67682
bytes_received_rset_header_packet	68915
bytes_received_rset_field_meta_packet	8416202
bytes_received_rset_row_packet	30690198
bytes_received_prepare_response_packet	0
bytes_received_change_user_packet	0
packets_sent_command	15279
packets_received_ok	1676
packets_received_eof	13522

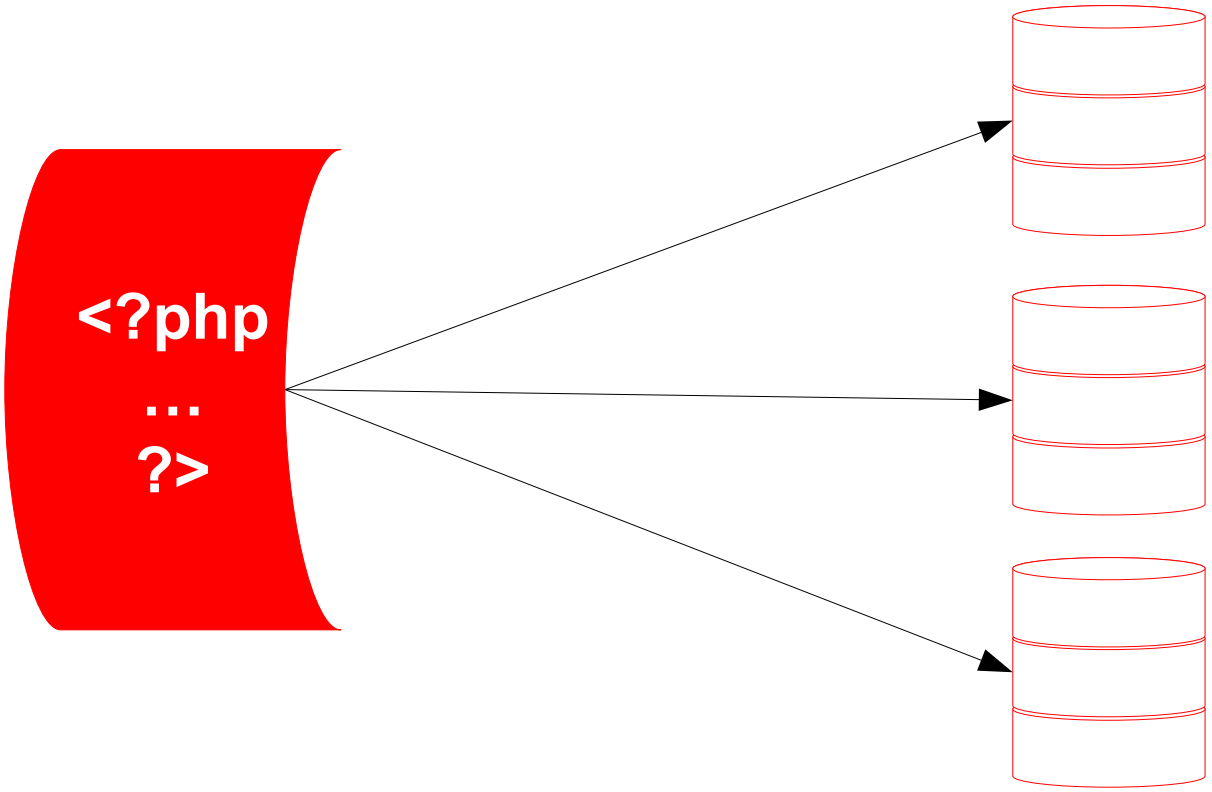
- Around 150 statistic values collected
- mysqli_get_client_stats (), mysqli_get_connection_stats()

Asynchronous Queries

```
$conn = new MySQLi(...);
$conn->query(
      "SELECT * FROM t WHERE ....",
      MYSQLI_ASYNC);

/* Do something */

mysqli_poll($links, $errors, $reject, 1);

/* Process query results */
```



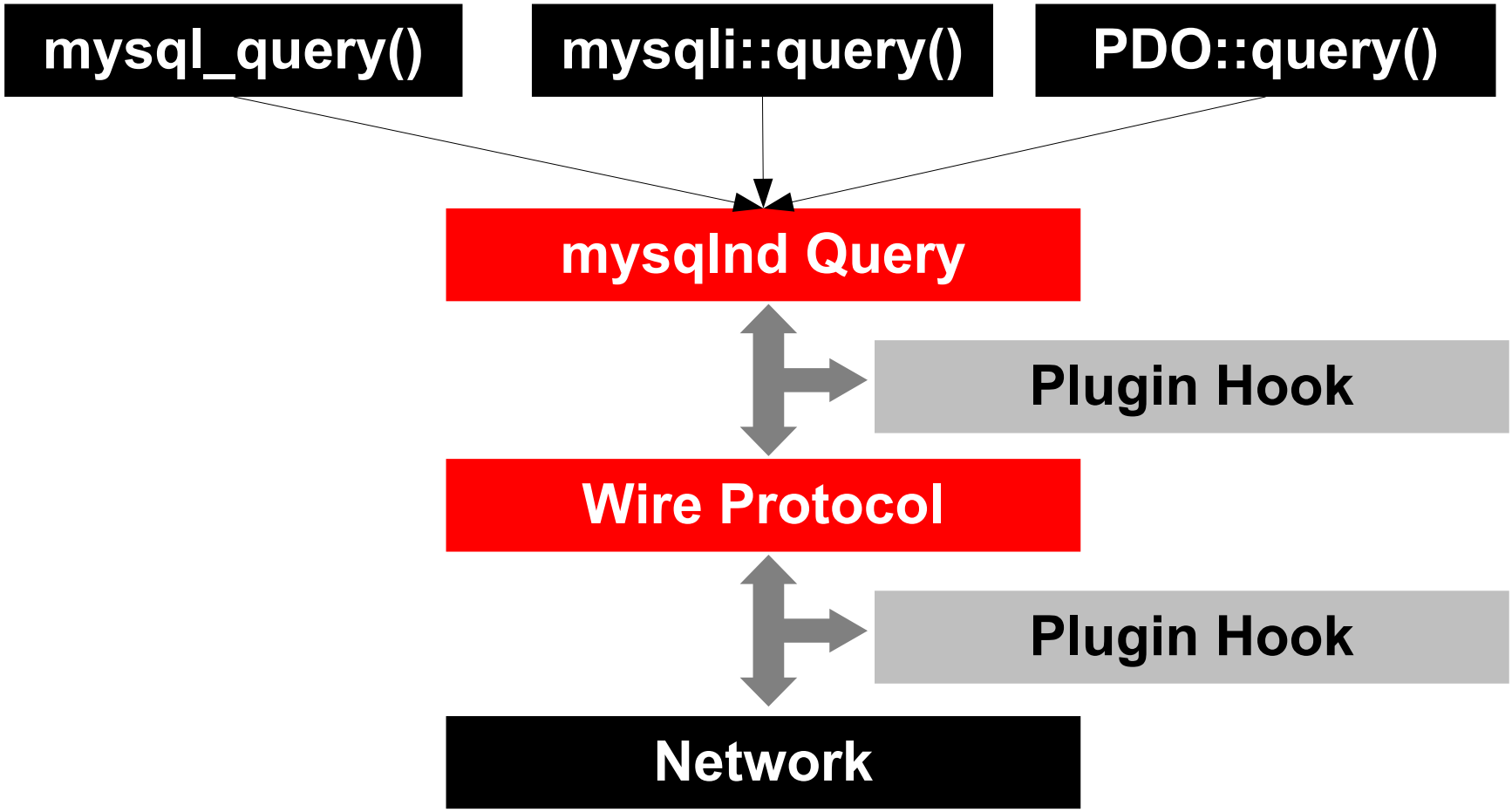
Sharding

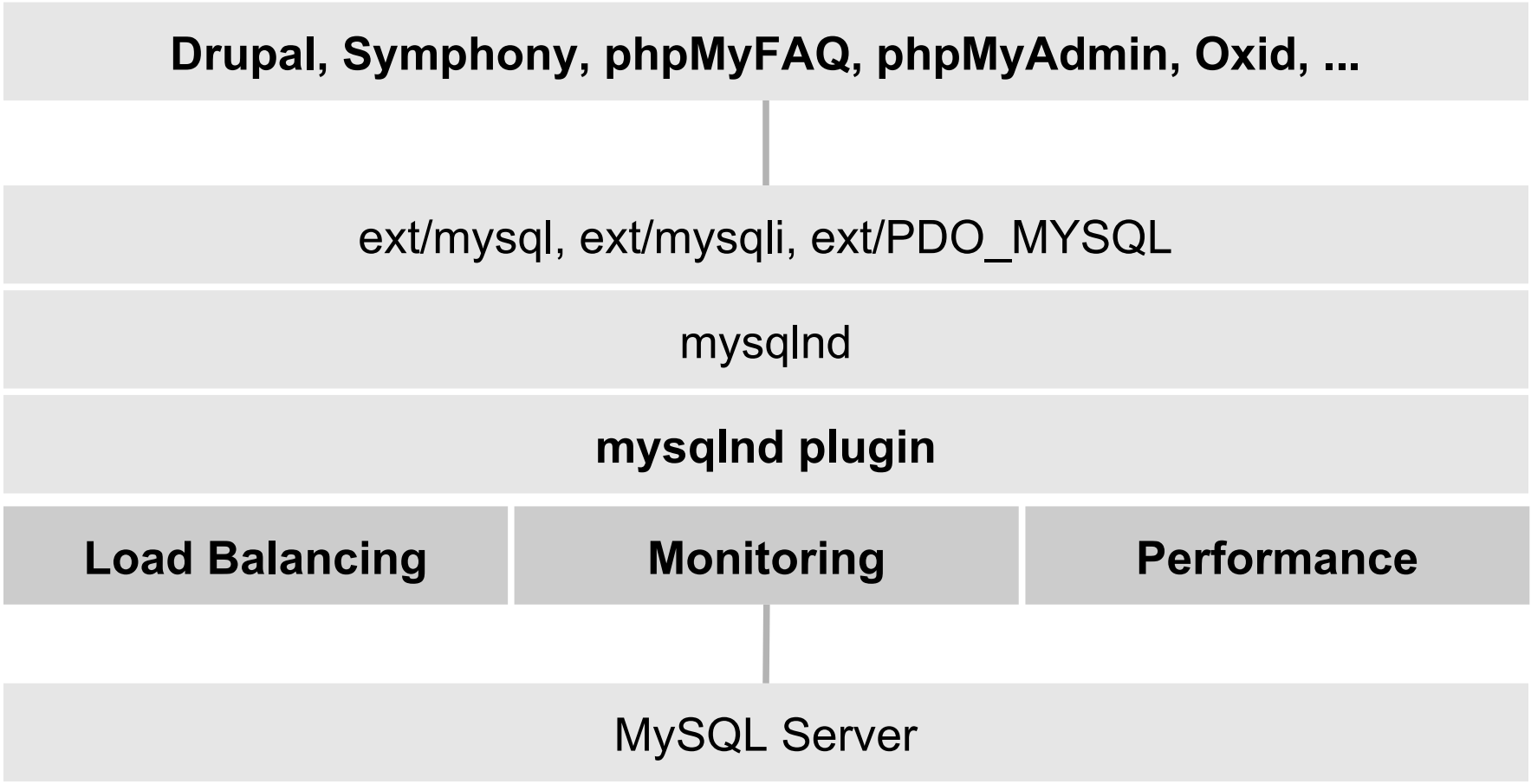


Sharding

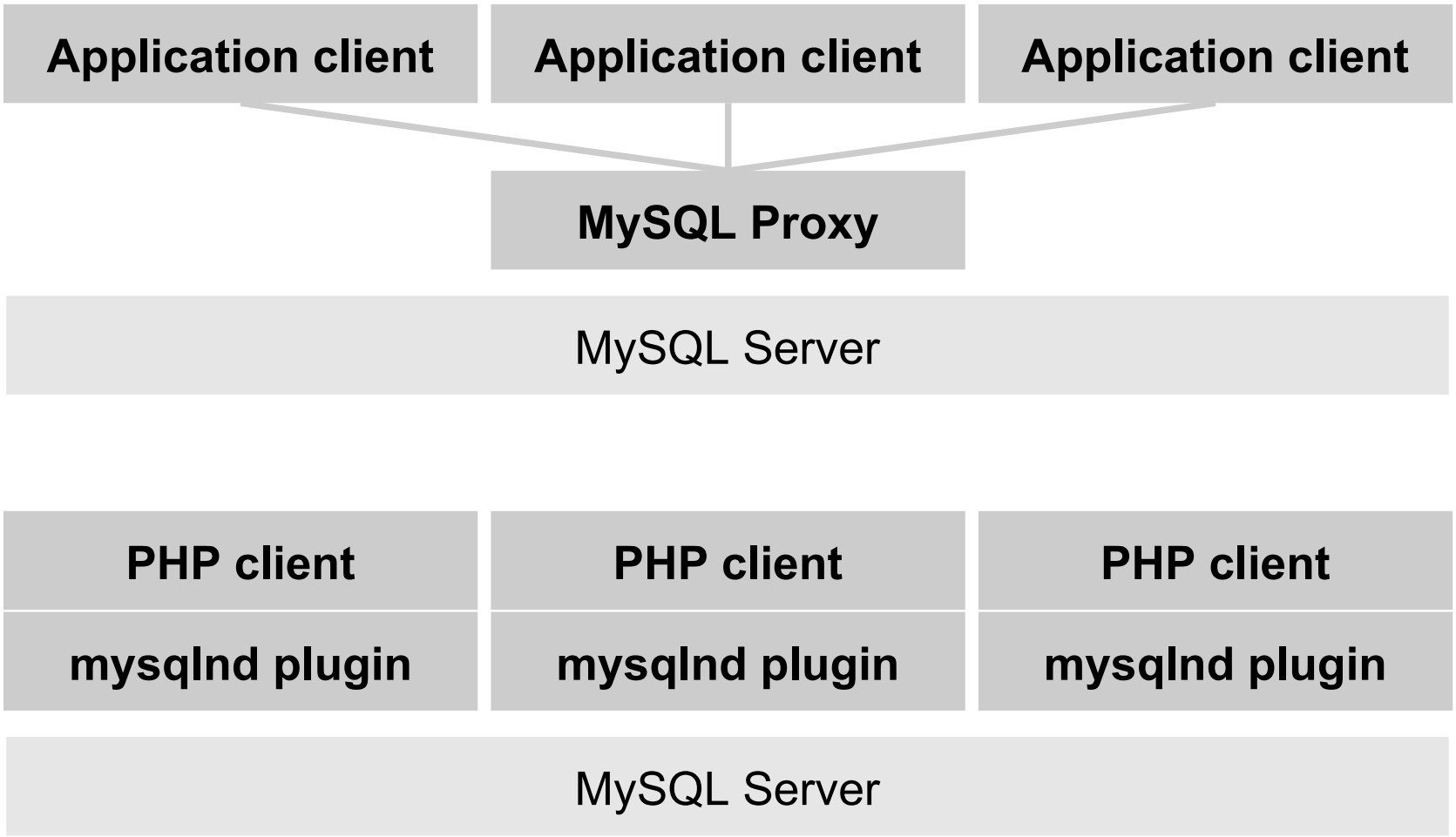
```
foreach ($all_links as $link)
    $link->query("SELECT 'test' ", MYSQLI_ASYNC);

$processed = 0;
do {
    $links = $all_links;
    if (!mysqli_poll($links, $errors, $reject, 1))  continue; /* TIMEOUT */

    foreach ($links as $link) {
        if ($result = $link->reap_async_query()) {
            print_r($result->fetch_row());
            mysqli_free_result($result);
            $processed++;
        }
    }
} while ($processed < count($all_links));
```

mysqlnd plugins

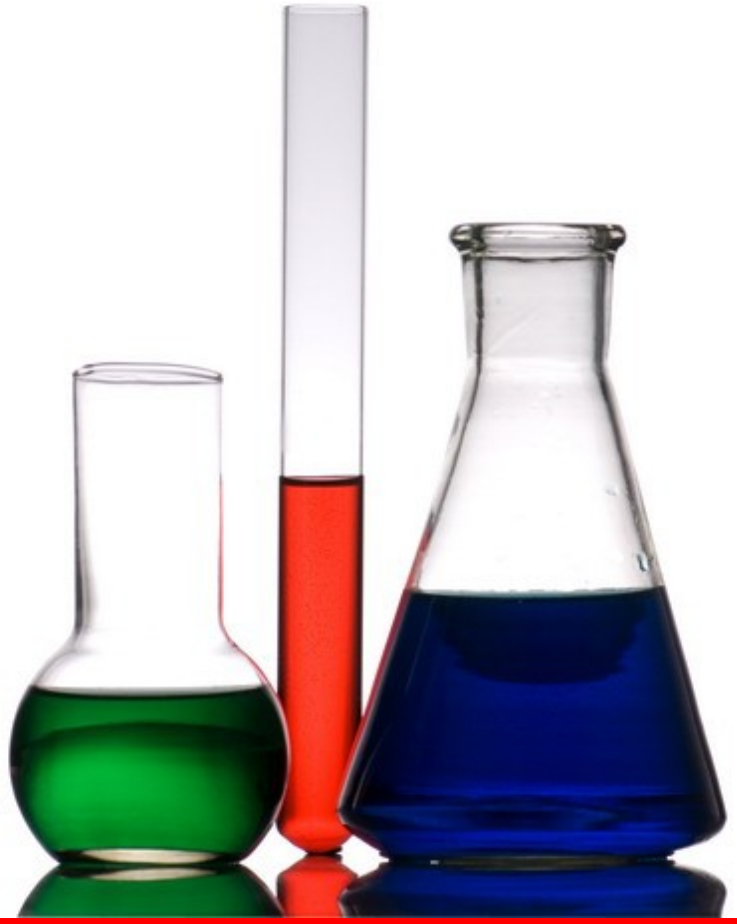



Drupal, Symphony, phpMyFAQ, phpMyAdmin, Oxid, ...
ext/mysql, ext/mysqli, ext/PDO_MYSQL
mysqlnd
mysqlnd plugin
Load Balancing
Monitoring
Performance
MySQL Server

Query Cache Plugin vs. MySQL Proxy

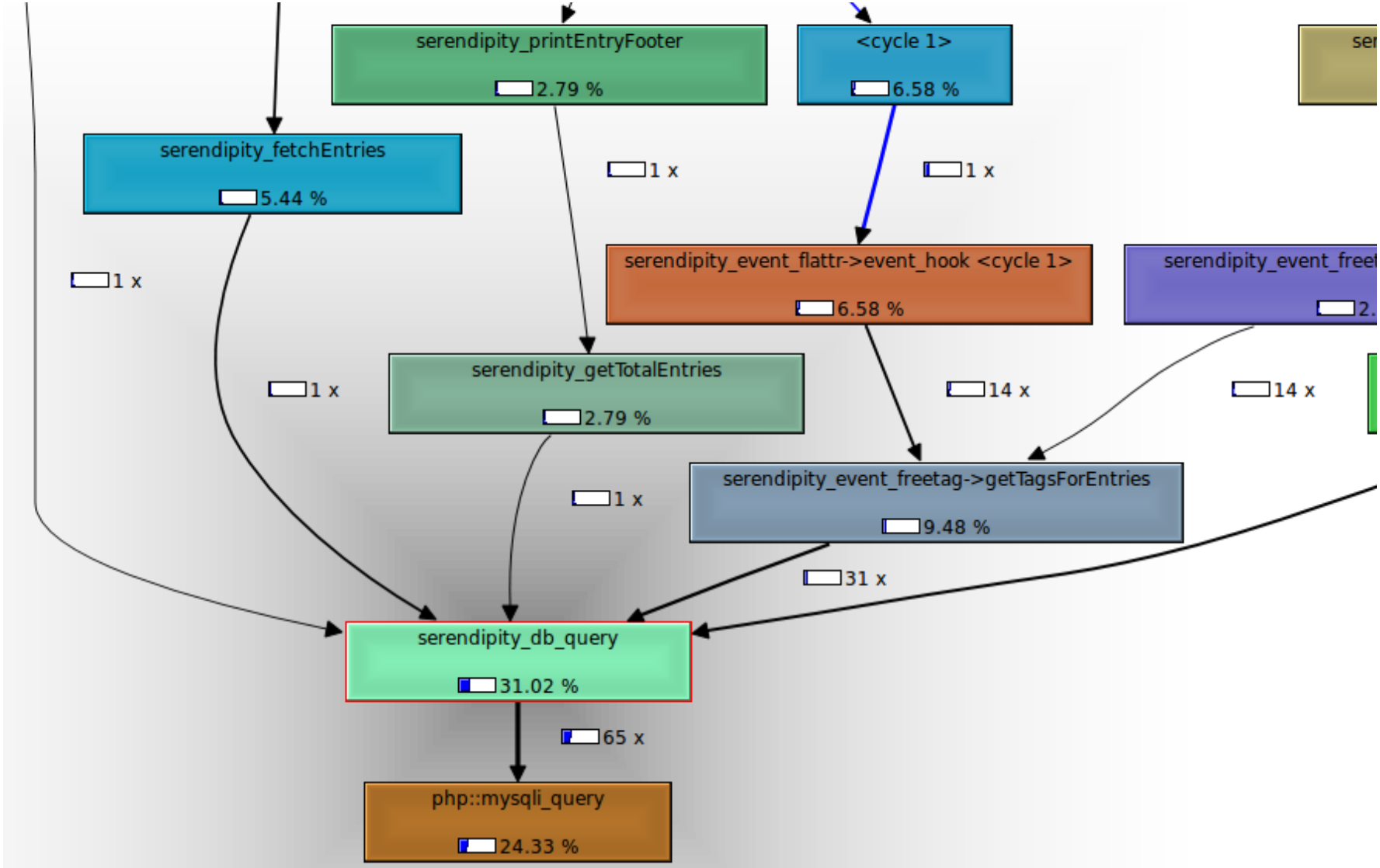


Experimental Extensions



- By Oracle:
  - mysqlnd_sip
  - mysqlnd_mc
  - mysqlnd_ms
  - mysqlnd_pscache
- By Community:
  - mysqlnd_uh (David Soria Parra / Mayflower GmbH)
- http://pecl.php.net/

The Database Is The Bottleneck



Caching!

“Traditional” Caches

- MySQL Server Cache
  - MySQL Query Cache
- Application-Level Cache
  - PEAR::Cache, Zend_Cache, …

MySQL Query Cache (Server-Side)

- ✔ Integrated with the MySQL Server
- ✔ No application changes
- ✔ Automatic invalidation
  - table based

- ✗ Needs network communication with server
- ✗ Needs server resources (memory, CPU)

Application-Level Caches

- ✔ Can cache higher level structures
  - Complete rendered HTML blocks
- ✔ Multiple backends
  - Can be shared over multiple servers (via memcache etc.)

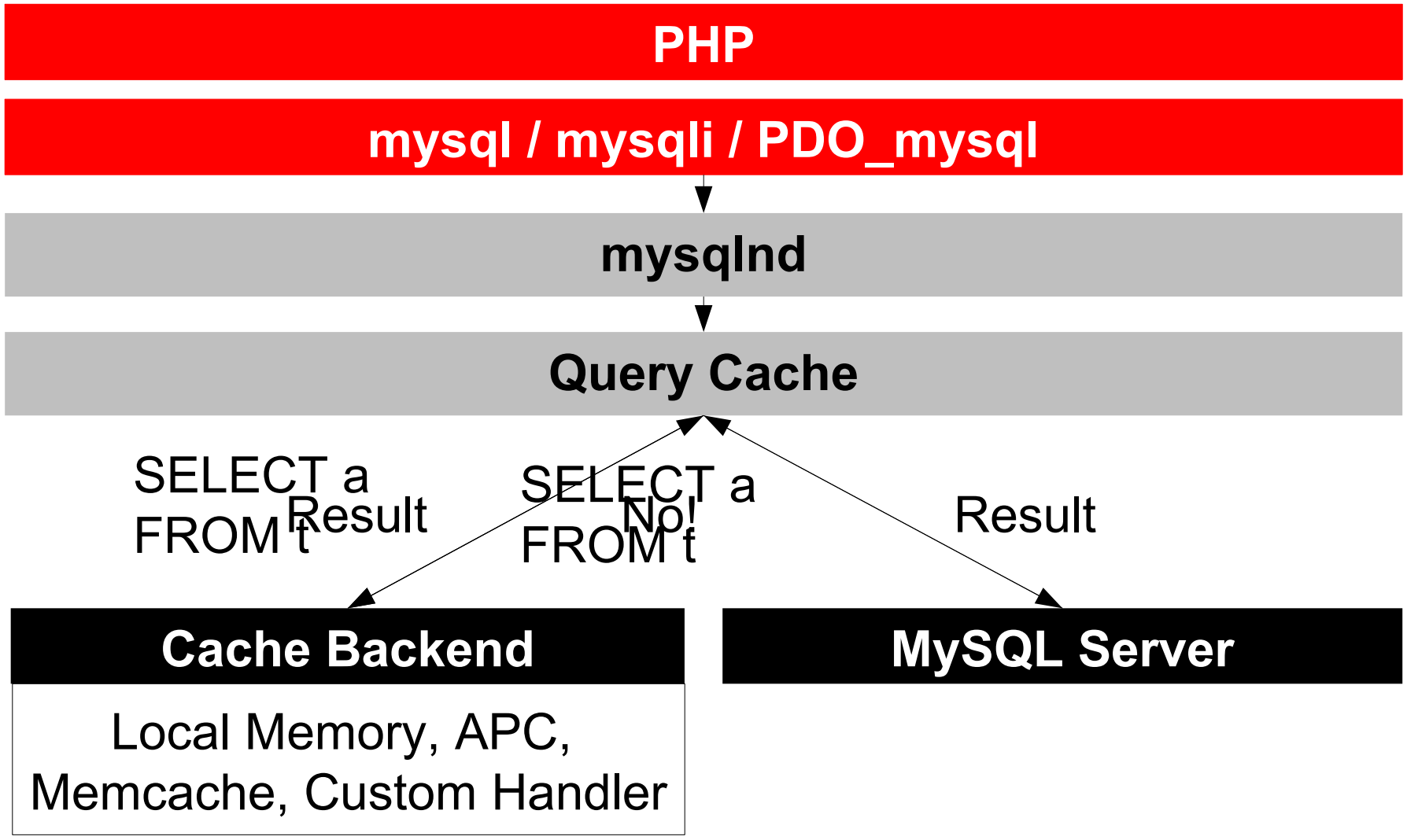
- ✗ No automatic invalidation
- ✗ Requires application changes

Introducing:
mysqlnd Client Side Cache

pecl install mysqlnd_qc-beta

Andrey Hristov
Ulf Wendel

mysqlnd Query Cache



Key Properties

- Transparent
  - PHP Extension hooking into mysqlnd
    - Works with ext/mysql, mysqli, pdo_mysql
- Pluggable storage handler
  - By default: local memory, APC, memcache, SQLite
  - PHP Userspace
- Invalidation via TTL
  - No automatic invalidation by server
  - Custom handlers may use custom invalidation logic

Transparent?

```
$mysqli = new mysqli($host, $user, $pw, $db);
$sql = “SELECT SLEEP(10) FROM table”;

$start = microtime(true);

$res = $mysqli->query($sql);
$res = $mysqli->query($sql);

$end = microtime(true);
echo $end - $start;
```

→ 20.019539117813

Transparent?

```
$mysqli = new mysqli($host, $user, $pw, $db);
$sql = sprintf("/*%s*/SELECT SLEEP(10) FROM table",
                MYSQLND_QC_ENABLE_SWITCH);

$start = microtime(true);

$res = $mysqli->query($sql);
$res = $mysqli->query($sql);

$end = microtime(true);
echo $end - $start;
```

→ 10.142804088593

Transparent!

mysqlnd_qc.cache_by_default = 1

! Usually you should NOT do this!

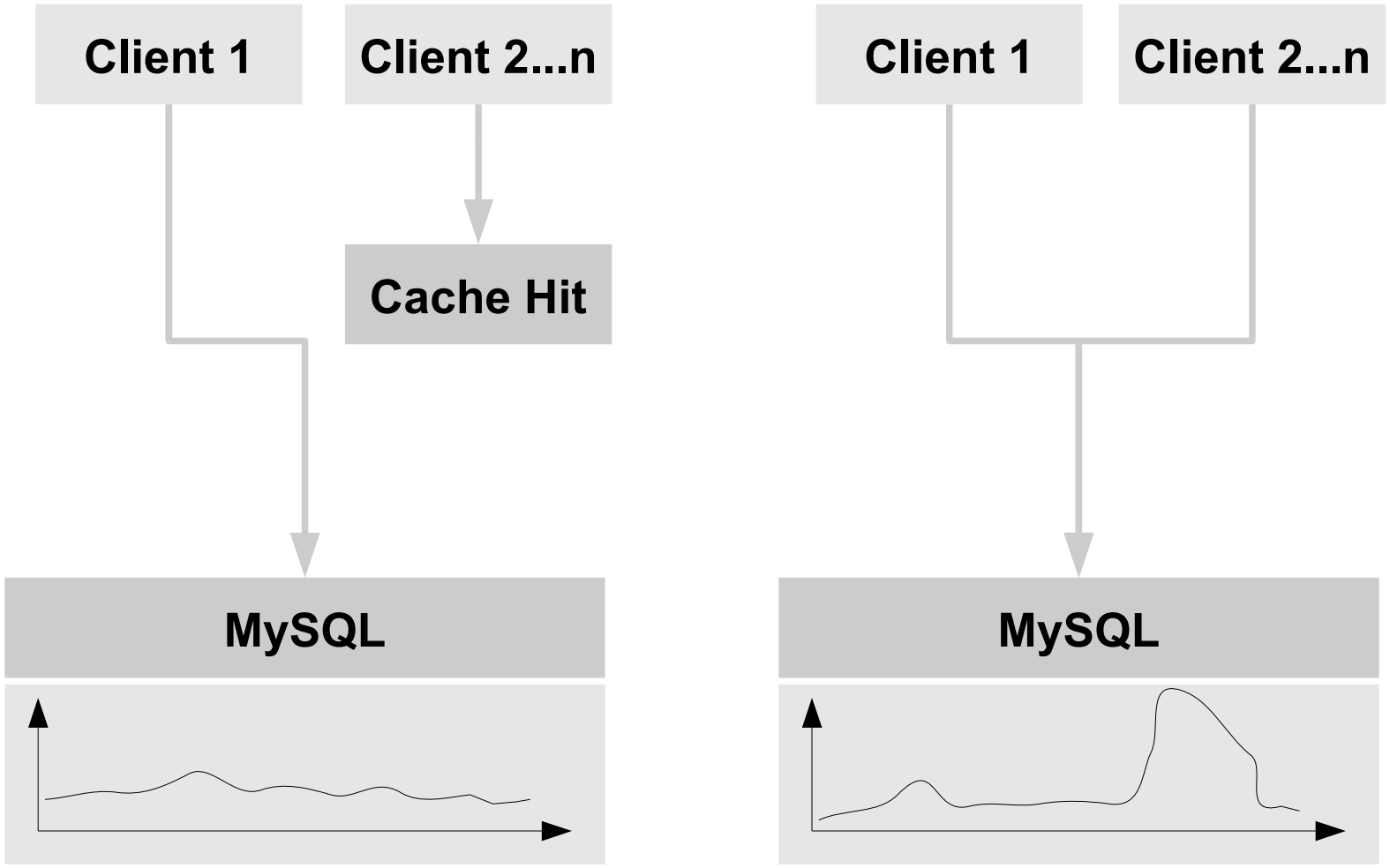
SQL Hints

- MYSQLND_QC_ENABLE_SWITCH
  - qc=on
- MYSQLND_QC_DISABLE_SWITCH
  - qc=off
- MYSQLND_QC_TTL_SWITCH
  - qc_ttl=

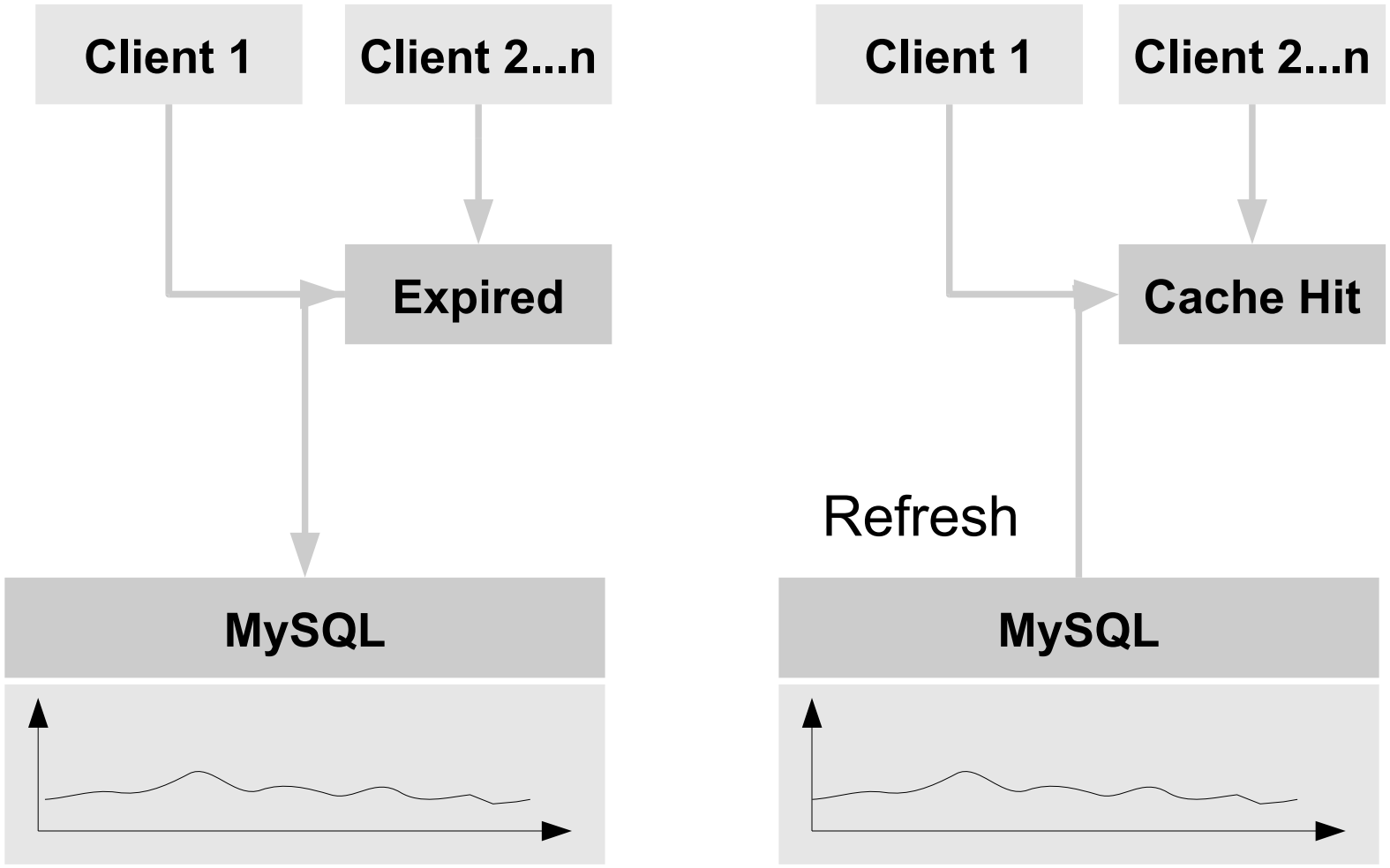
Storage Handlers

- **Storage**
  - Scope: request, process, machine, multi-machine
  - Location: distance to cache
  - Replacement strategy
  - Slam defense strategy
- **Decide what to cache**
  - is_select() - detect SQL hints
- **Extended statistics**
  - Storage statistics, traces, timings

Cache Expiry



Slam Defense



Core Statistics

- Collected by: mysqlnd_qc core
  - php.net/manual/en/function.mysqlnd_qc_get_core_stats.php
- Scope: process
  - Process: mysqlnd_qc_get_core_stats()
  - Aggregated values from all PHP MySQL APIs
- Contents: wide range
  - Cache usage and efficiency
  - Network related
  - Timings

Query Statistics and Backtraces

- Collected by: mysqlnd_qc core
  - php.net/manual/en/function.mysqlnd_qc_get_query_trace_log.php
- Scope: process
  - mysqlnd_qc_get_query_trace_log()
  - mysqlnd_qc_get_normalized_query_trace_log()
- Contents
  - Origin - backtrace
  - Timings

Storage Handler Statistics

- Collected by: storage handler
  - php.net/manual/en/function..mysqlnd_qc_get_cache_info.php
- Scope: cache entry
  - Depends on storage handler scope
  - Aggregated values from all PHP MySQL APIs
- Contents: none or assorted
  - Depends on storage handler support
  - APC: timings, hit ratio, result set size
  - Default: APC plus result set meta data

User-Defined Storage Handler

- Procedural
  - php.net/manual/en/function.mysqlnd_qc_set_user_handlers.php
  - Callback functions
- Object oriented
  - slideshare.net/nixnutz/mysqlnd-query-cache-plugin-userdefined-storage-handler
  - Interface mysqlnd_qc_handler
- Extending mysqlnd_qc_handler_default

```
<?php
/* auto_prepend_file = /path/to/prepend.php */

class my_qc extends mysqlnd_qc_handler_default
{
  public function is_select($query)
  {
    if (preg_match("@from employees where@ism", $query))
    {
      /* cache query for mysqlnd_qc.ttl seconds */
      return true;
    }
    return parent::is_select($query);
  }
}

$qc = new my_qc();
mysqlnd_qc_change_handler($qc);
?>
```

Resources

- php.net/mysqlnd_qc
  - Installation, Examples, Functions
- php.net/mysqlnd
  - Introduction, Changes, C plugin API
- slideshare.net/nixnutz/presentations
  - QC Basics
  - QC User defined storage handler
  - QC Statistics
  - QC Benchmark impressions

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

SOFTWARE. HARDWARE. COMPLETE.


ORACLE®