ProxySQL 2.7 → 3.0: аудит кеша в PmaControl примерно в 4 раза быстрее

Aurélien LEQUOY · September 27, 2026

PMACONTROL PROXYSQL BENCHMARK MARIADB MYSQL OBSERVABILITY

PmaControl BENCHMARK NOTES

ProxySQL 2.7 → 3.0

Same complete cache audit. Same two Admin queries.

ProxySQL 2.7.3
8.112 ms/audit

ProxySQL 3.0.5
2.015 ms/audit

−75.2% mean elapsed time **4.03× faster**

150 timed audits per version · Ubuntu 24.04.4 · 2 vCPU · 4 GiB · PHP 8.4.26

Client-observed Admin workload: SQL + loopback + fetch + PHP. Query-routing throughput is outside scope.

Приятное улучшение, обнаруженное во время бенчмарков PmaControl: одна и та же полная проверка кеша занимала в среднем **8,112 и 2,015 миллисекунды** на протестированных сборках ProxySQL 2.7.3 и 3.0.5 соответственно. Это **на 75,16% меньше времени**, или сокращение средней длительности в **4,03 раза**.

Публикуем цифры, их определения, запросы и методику, чтобы результат можно было точно интерпретировать. Нагрузка — проверка эффективности кеша PmaControl через интерфейс **Admin** ProxySQL.

Результаты при одинаковом объёме работы

Метрика	ProxySQL 2.7.3	ProxySQL 3.0.5	Снижение времени
Среднее, ms/audit	8.112204	2.014760	75.16 %

Метрика	ProxySQL2.7.3	ProxySQL3.0.5	Снижениевремени
Медиана средних по сериям, ms/audit	7.895812	2.053810	73.99 %
Максимум средних по сериям, ms/audit	11.914402	2.919234	75.50 %
Пик выделенной памяти PHP, МиБ	4	4	без изменений

Время в таблице указано в **ms/audit** — миллисекундах на аудит; память имеет отдельную единицу. Медиана и максимум рассчитаны по **15 средним значениям серий из 10 аудитов**. Максимум не означает самый медленный отдельный вызов или перцентиль задержки. Файлы данных сохраняют исходную точность; инфографика округляет время до трёх знаков после запятой.

Пик 4 МиБ относится к памяти, выделенной процессу PHP. Это не потребление памяти сервером ProxySQL.

PmaControl

BENCHMARK NOTES / 27 SEP 2026

ProxySQL 2.7 → 3.0

A nice improvement during PmaControl cache-audit benchmarks.

−75.2%
MEAN ELAPSED TIME PER AUDIT

4.03×
faster complete cache audits

Mean time · complete cache audit

ms/audit · lower is better

Supporting measurements

	2.7.3	3.0.5	UNIT
Median batch mean	7.896	2.054	ms/audit
Maximum batch mean	11.914	2.919	ms/audit

Median / maximum across 15 batch means; each batch averages 10 audits.

What “ms/audit” measures

One complete cache audit = two read-only ProxySQL Admin queries.

- 01 SQL execution
- 02 Loopback + fetch
- 03 PHP audit logic

Client elapsed time. Connection setup, process startup, HTTP and UI rendering are outside the timer.

The benchmark setup

Ubuntu 24.04.4 · 2 vCPU · 4 GiB RAM · PHP 8.4.26

150 timed complete audits per version

15 batches × 10 audits · 3 warmups per batch

Same audit code and SQL · 1 + 7 rows returned per audit

PHP allocated peak: 4 MiB for both versions

Scope: this cache-audit workload on a shared test VM.
These are client-observed Admin timings. Query-routing throughput and standalone server execution time were outside this benchmark.

Source: PmaControl PR #6389 · fixed complete audit on both versions
Builds: 2.7.3-12-g50b7f85 / 3.0.5-60-g7e9e009

Nice improvement, ProxySQL.

Что означает «ms/audit»

Под аудитом здесь понимается **один полный вызов** `QueryCacheEffectiveness::run()`, а не весь набор проверок сервера. Он выполняет два SELECT только для чтения: первый получает активные правила кеширования и счётчики совпадений, второй — счётчики кеша и настроенный размер. Тестовые данные возвращают **одну строку**, затем **семь строк**.

Таймер охватывает всю PHP-проверку: выполнение нативных SQL-запросов, обмен через loopback, получение результатов и обработку в PmaControl. Установление соединения, запуск процесса PHP, HTTP и отрисовка интерфейса исключены. Это время со стороны клиента, без выделения внутреннего времени исполнения ProxySQL.

Ниже приведены оба запроса с форматированием для удобства чтения. Их логика одинакова на обеих версиях.

```
SELECT r.rule_id, r.cache_ttl, r.digest,
       r.match_digest, r.match_pattern,
       COALESCE(h.hits, 0) AS hits
FROM runtime_mysql_query_rules r
LEFT JOIN stats_mysql_query_rules h
       ON h.rule_id = r.rule_id
WHERE r.active = 1 AND r.cache_ttl > 0
ORDER BY r.rule_id;
```

```
SELECT Variable_Name AS name, Variable_Value AS value
FROM stats_mysql_global
WHERE Variable_Name IN (
    'Query_Cache_count_GET', 'Query_Cache_count_GET_OK',
    'Query_Cache_count_SET', 'Query_Cache_Memory_bytes',
    'Query_Cache_Entries', 'Query_Cache_Purged'
)
UNION ALL
SELECT variable_name, variable_value
FROM global_variables
WHERE variable_name = 'mysql-query_cache_size_MB';
```

Поле `stats_mysql_query_rules.hits` считает совпадения правила, а не попадания в кеш, относимые к этому правилу. Счётчики кеша накопительные и сами по себе не показывают недавний процент попаданий. Значение `mysql-query_cache_size_MB`, прочитанное из MAIN, — настроенная цель очистки, а не жёсткий лимит; эта проверка не подтверждает значение runtime.

Методика бенчмарка

Исходный бенчмарк проводился на **общей тестовой VM с Ubuntu 24.04.4 LTS**, **2 vCPU**, **4 ГиБ RAM** и **PHP 8.4.26**. Использовались loopback-соединения и следующие нативные сборки ProxySQL:

- `2.7.3-12-g50b7f85` для серии 2.7;
- `3.0.5-60-g7e9e009` для серии 3.0.

Для каждой версии опубликованная выборка содержит **15 серий по 10 измеряемых аудитов**, после **3 прогревочных вызовов на серию**: 150 полных аудитов на версию, 300 суммарно и 600 SELECT. PHP-код проверки и строки SQL одинаковы; хеш кода и число возвращённых строк проверяются в каждой серии и вызове.

В исходном эксперименте старый и исправленный варианты PmaControl чередовались в 15 парах AB/BA, с отдельным процессом PHP для каждого варианта. В этой публикации выбран только исправленный вариант. В каждом процессе сначала запускался ProxySQL 2.7.3, затем 3.0.5: **порядок версий ProxySQL не чередовался**. 600 вызовов и 900 SELECT всего исходного эксперимента, включая старый вариант, не являются размером представленной здесь выборки.

Почему сравниваем полную проверку

Бенчмарк сопровождал функциональное исправление: допустимое правило со знаковым идентификатором `rule_id=-1` и TTL 5 000 мс раньше завершало проверку после первого чтения. Проверка ошибочно сообщала, что статистика недоступна; второй запрос не выполнялся.

Сравнение такого неполного результата с исправленной проверкой означало бы разный объём работы. Поэтому мы используем **одинаковый исправленный код на обеих версиях ProxySQL**, с двумя чтениями и полными наблюдениями. Опубликованный выигрыш сравнивает сборки ProxySQL в этом сценарии и не приписывается исправлению валидации PmaControl. Существующие измерения были перепроверены; новый бенчмарк для статьи не запускался.

Какие выводы позволяет сделать улучшение

В этой среде среднее время, медиана средних по сериям и максимум средних ниже на протестированной сборке 3.0. Пиковый объём выделенной PHP-памяти не изменился. Это измеримое улучшение для задачи мониторинга.

Результат относится к общей VM, небольшим наборам результатов и конкретной проверке. Он не подтверждает пропускную способность прикладных запросов, задержку frontend-маршрутизации или такой же выигрыш при высокой конкуренции. Тест не выделяет серверные фазы prepare/step/copy/finalize, не доказывает бюджет SQL менее 1 мс и не оценивает производственную ёмкость или долговечность данных. Доверительный интервал не установлен, порядок версий фиксирован.

В этих границах **полная проверка кеша занимала в среднем примерно в четыре раза меньше времени**. Отличное улучшение, ProxySQL!

Пересчёт результатов

Публичные данные содержат 30 выбранных серий, 300 отдельных длительностей, количество строк результатов и метаданные методики. Внутренние пути и диагностические сообщения удалены; времена сохранены без изменений. Скрипт Python использует только стандартную библиотеку и пересчитывает таблицу, а не запускает бенчмарк заново.

Снижение времени вычисляется как `100 × (1 – time_3.0 / time_2.7)`, отношение — как `time_2.7 / time_3.0`. Размер всех серий одинаков. Хеш ниже идентифицирует реально загруженный код проверки; commit — ревизия PR, содержащая эту реализацию.

```
curl -fsSLO https://pmacontrol.com/downloads/proxysql-2.7-vs-3.0/benchmark-data.json
curl -fsSLO https://pmacontrol.com/downloads/proxysql-2.7-vs-3.0/analyze.py
python3 analyze.py benchmark-data.json
```

```
PmaControl revision:
b4e7b4c6ca72cf27077ddbdb1f2c89af05800b5a
QueryCacheEffectiveness helper SHA-256:
c47a169935f665259554bd0fbaa85e90af9cbcd3983bf7ad3548e2caaf63d4dc
```

Графика, публикация и источники

- Полная инфографика PNG, 1600 × 2000
- Редактируемый SVG со всеми элементами
- Данные бенчмарка JSON
- Скрипт пересчёта на Python
- Публикация для LinkedIn на английском
- Архив графики и скриптов генерации

Графика подготовлена на английском для распространения; статья полностью включает пояснения и измерения. Кнопка PDF на странице также экспортирует статью с изображениями.

Исходная работа: PmaControl PR #6389 и задача публикации #6390. Ссылки Forgejo требуют доступа к репозиторию; файлы выше доступны публично.