Кеш и индексы есть, а страница работает медленно: расследование MariaDB в PmaControl

Aurélien LEQUOY · October 9, 2026

PMACONTROL MARIADB PERFORMANCE SQL BENCHMARK ARCHITECTURE



«Чертежи уже находятся в кеше. Почему страница „Архитектура“ по-прежнему отображается несколько секунд?»

Этот вопрос привел нас к очень короткому исправлению в PmaControl. **Отсутствующего индекса не было. Также не было необходимости пересчитывать рисунки.** MariaDB выбрала дорогостоящий способ поиска генерации графиков для отображения.

Вот метрики, SQL и ограничения этого сравнения. Мы различаем **клиента**, который представил проблему, и тестовый экземпляр, который уже быстро выполнял первоначальный запрос.

Обновление от 10 октября 2026 г. Ниже добавлены измерения SQL, записанные в 13:29. Время скачивания повторно измерено около 13:45 с сервера в локальной сети, поскольку

соединение, использованное для замеров 9 октября, было нестабильным. Для каждой серии сохранены собственные окружение, число графиков и область измерения: SQL, HTTP-скачивание или загрузка в Chromium.

Объяснение на примере коробки с чертежами

Представьте себе библиотеку, полную коробок. В каждой коробке содержатся чертежи одного поколения топологии. Рисунки уже готовы: это наш SVG-кеш.

Чтобы показать страницу, приложению всё ещё нужно найти нужную коробку, достать из неё чертежи и расположить их на экране. Каталог библиотеки уже существует — это индекс SQL.

Но наличия каталога недостаточно. В медленном случае MariaDB решила перебрать миллионы записей, чтобы подготовить промежуточный список совпадений. Для интересующей нас проверки достаточно найти **единственное совпадение**.

Исправление явно выражает эту проверку. Возвращаются те же чертежи, а поиск избегает дорогостоящего обходного пути.

Что уже было кэшировано

SVG сохраняются в `dot3_graph`. Таблица `dot3_cluster` связывает их с поколениями топологии. Страница уже получает все графики с помощью **одного SQL-запроса**, который возвращает, в частности:

```
SELECT dg.id, dg.svg, dg.height, dg.width,
       (dg.height * dg.width) AS area,
       dc.date_inserted AS date_refresh
FROM dot3_cluster dc
JOIN dot3_graph dg ON dg.id = dc.id_dot3_graph
-- Условие выбирает поколение, готовое к отображению.
```

Браузер получает чертёж, его высоту, ширину и площадь. Код в `App/Webroot/js/Architecture/topology.js` уже размещает чертежи в браузере. Исправление не меняет этот алгоритм.

Кэш позволяет избежать создания рисунков при каждом посещении. Он не исключает поиска, передачи и отображения рисунков. Этот кеш SVG отделен от кеша запросов MariaDB и HTTP-кеша браузера.

Нет, индекс не отсутствовал

Медленный экземпляр уже имел этот уникальный индекс, что подтверждает `SHOW INDEX FROM dot3_cluster`:

```
-- Существующий индекс в виде определения:
UNIQUE KEY id_dot3_information_2
    (id_dot3_information, id_dot3_graph)
```

Поиск фильтрует по `id_dot3_information`, первому столбцу индекса. Поэтому индекс позволяет найти кластеры заданного поколения.

Проблема была в **плане выполнения**, то есть в стратегии, выбранной MariaDB для выполнения SQL. Исходная версия использовала `EXISTS`, чтобы проверить наличие хотя бы одного кластера у поколения. На этом экземпляре такой выбор приводил к материализации: MariaDB готовила промежуточный результат, просматривая около **4,33 миллиона исторических записей**.

Это не означает, что браузер получил четыре миллиона графиков: он получил **61 график**. Лишняя работа происходила внутри базы данных до получения полезного результата. Число 4,33 миллиона описывает порядок объёма сканирования, выявленного при первой диагностике, а не число строк, переданных в PHP. Измерение от 10 октября показывает около **5,2 миллиона строк** в таблице этого экземпляра; поэтому объёмы данных исходной статьи и новой серии не полностью совпадают.

Индекс можно использовать при очень широком сканировании. Поэтому наличие индекса само по себе не гарантирует короткого доступа.

Конфигурация и таблицы MariaDB, InnoDB/RocksDB

Обе среды используют **MariaDB 11.8.9**, но разные пакеты: **11.8.9-deb13 у клиента**, согласно его измерению от 10 октября, и **11.8.9-MariaDB-deb12-log на тестовом экземпляре**, что проверено непосредственно на сервере. Клиент использует PHP 8.4 для

воспроизведения SQL, а тестовый экземпляр — FrankenPHP / PHP 8.5.9.

На диаграмме ниже показаны **16 таблиц приложения, используемые 16 запросами на странице**, помимо трех таблиц в запросе SVG. Полезные столбцы, ключи и механизмы взяты из фактической диаграммы тестового экземпляра, отмеченной **10 октября в 15:06**. Проверки наличия таблиц также запрашивают `information_schema.tables`, представление метаданных, которое не является таблицей приложения, которую можно классифицировать как InnoDB или RocksDB.

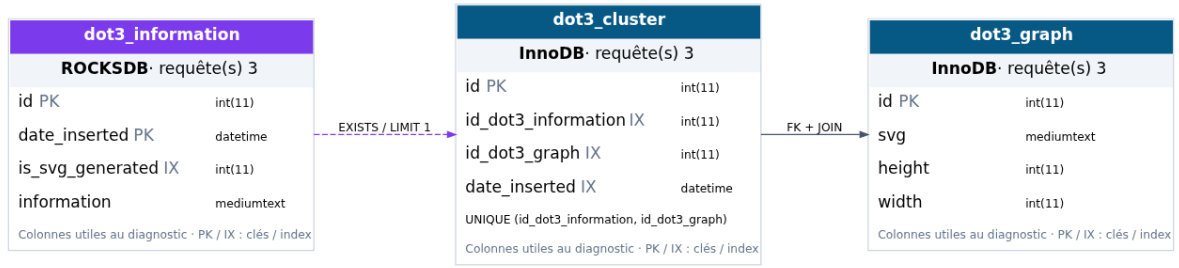
Tables de la page Architecture · partie 1/2 · instance de test

7 tables InnoDB (bleu) · 1 table RocksDB (violet) · relevé du 10 octobre à 15 h 06

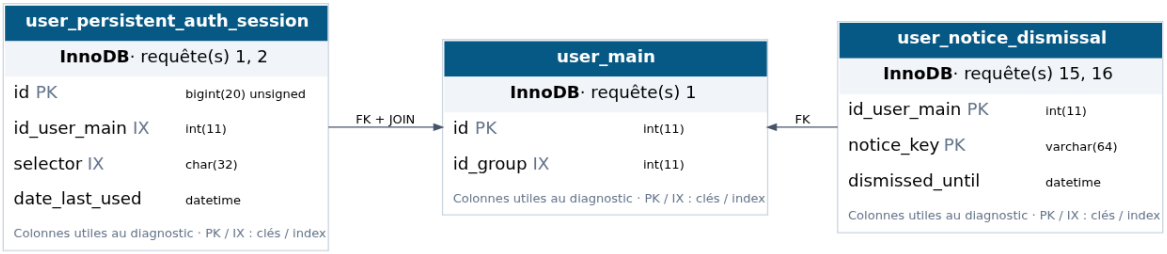
Trait plein : clé étrangère déclarée · Pointillé : jointure logique du SQL

Colonnes utiles au diagnostic ; schéma complet des 16 tables disponible en SVG.

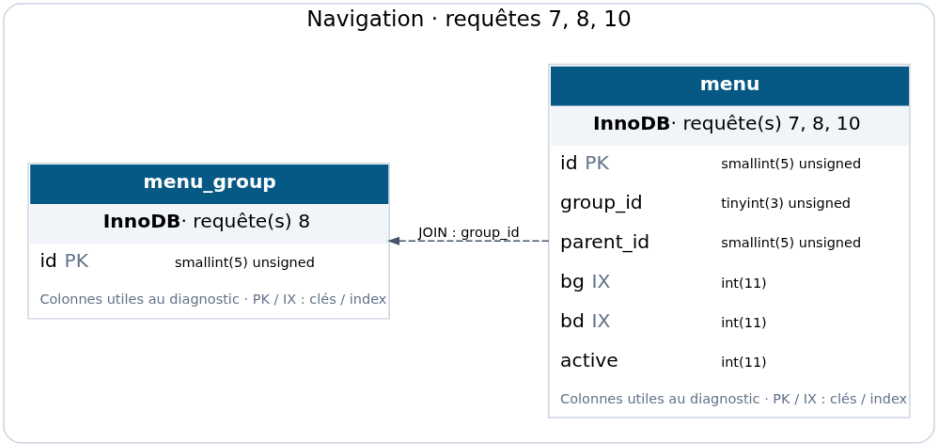
Sélection et lecture des SVG · requête 3



Session et masquage de bandeau · requêtes 1, 2, 15, 16



Navigation · requêtes 7, 8, 10





Открытая векторная диаграмма, большая. Сплошная линия указывает на объявленный внешний ключ; пунктирная линия указывает логическое соединение, используемое SQL. Таблицы без отображаемой ссылки запрашиваются отдельно. На рисунке показаны связи, а не порядок выполнения запроса.

Таблица SVG-запроса на тестовом экземпляре	Наблюдаемый движок хранения
`dot3_information`	**RocksDB**
`dot3_cluster`	**InnoDB**
`dot3_graph`	**InnoDB**

Остальные тринадцать таблиц страницы используют InnoDB. Полная схема выше и скачиваемый перечень движков и размеров содержат подробности.

Таким образом, SVG-запрос проходит через **два движка хранения** на тестовом экземпляре. Индекс доступности `dot3_information` относится к RocksDB, а уникальный индекс `dot3_cluster` и данные SVG в `dot3_graph` — к InnoDB. Исправление сохраняет эти движки и индексы. Серия сравнивает две формулировки SQL и не измеряет перенос таблиц из InnoDB в RocksDB. В измерениях клиента движки не указаны, поэтому мы не приписываем его среде схему тестового экземпляра.

Анонимизированная конфигурация `.cnf` тестового экземпляра объединяет серверные разделы подключаемых файлов в порядке их загрузки. Сетевые идентификаторы и секреты удалены. Поскольку последующие файлы могут переопределять предыдущие значения, фактически считанные глобальные переменные показывают активные настройки: **2 ГиБ буферного пула InnoDB**, **2 ГиБ кэша блоков RocksDB**, **не более 128 подключений**, **64 МиБ для `tmp_table_size` и `max_heap_table_size`** и **отключённый кэш запросов**.

Параметры собраны в **15:14**, после измерений в 13:45; некоторые переменные могут меняться динамически. Ресурс `.cnf` описывает наблюдаемую конфигурацию и её переопределения с удалёнными чувствительными значениями; его нельзя применять напрямую. Измерения клиента не содержат его конфигурацию `.cnf`.

Модификация SQL до и после

Изменение находится в `App/Library/Dot3LatestGeneration.php`, в разделе выбора последнего поколения.

До:

```
AND EXISTS (
    SELECT 1
    FROM dot3_cluster dc_ready
    WHERE dc_ready.id_dot3_information = di.is_svg_generated
)
```

После:

```
AND (
    SELECT dc_ready.id_dot3_information
    FROM dot3_cluster dc_ready
    WHERE dc_ready.id_dot3_information = di.is_svg_generated
    LIMIT 1
) IS NOT NULL
```

Почему здесь эквивалентно? Внешний запрос уже накладывает `di.is_svg_generated > 0`. Если существует соответствующий кластер, подзапрос возвращает этот положительный идентификатор. В противном случае скалярный подзапрос без результатов возвращает `NULL`. Таким образом, условие всегда проверяет наличие кластера для этого поколения.

Скалярный подзапрос позволяет избежать плана материализации, наблюдаемого с `EXISTS`. Движок может искать поколение в индексе и останавливаться на первом совпадении. Резервные решения, используемые в случае отсутствия готовой генерации, остаются прежними.

Это не правило, утверждающее, что `EXISTS` работает медленно. В другом случае старый запрос уже использовал эффективную стратегию `FirstMatch`. Переписывать все `EXISTS` было бы плохим выводом: придется изучать и измерять план каждого запроса.

Клиент: новые измерения SQL от 10 октября

Измерения у **клиента** около **13:29** сравнивают по десять запусков каждого варианта на производственных данных. Один прогревочный запуск исключён, порядок до/после чередуется. Производственный код уже содержит исправление: «до» здесь означает сравнительный запуск старого запроса `EXISTS`, а «после» — скалярный подзапрос с `LIMIT 1`.

SQL-измерения	**До (EXISTS)**	**После (LIMIT 1)**
SVG-запрос, минимум–максимум	4 434,17–5 297,38 ms	2,69–6,20 ms
SVG-запрос, медиана	**5 044,59 ms**	**3,32 ms**
Всего 16 запросов, медиана	5 068,52 ms	12,92 ms
Возвращенные графики	61	61
Объем SVG	995 307 байт	995 307 байт

SQL-измерения	До (`EXISTS`)	После (`LIMIT 1`)
Счётчик чтений индекса в измерении	5 199 176	3

Время доминирующего запроса сокращается с **5,04 секунды до 3,32 мс**, то есть в этом измерении он работает примерно в **1520 раз быстрее**. Все 61 графика идентичны до и после исправления. Текущий объём отличается на 130 байт от первой диагностики, где было 995 437 байт; это разные даты измерения, а оба варианта новой серии возвращают одинаковое содержимое.

План `EXPLAIN` до исправления **оценивает 4 406 092 записи** для materialization, а после — **81 запись** для зависимого подзапроса с индексом. Эти оценки не являются числом фактически выполненных чтений. Счётчик чтений индекса показывает **5 199 176 до и 3 после**; в тот момент `dot3_cluster` содержала **5 198 990 строк**. Оценка плана, фактические чтения и переданные графики — три разных показателя.

Медианы остальных 15 запросов остаются ниже 10 мс. Некоторые колеблются: например, счетчик сигналов тревоги меняется от 9,21 до 0,85 мс, а иногда и раньше достигает 15,55 мс. Этот результат не демонстрирует, что исправление ускоряет подсчет: загрузка сервера составляла около 6, и эти запросы следовали сразу за пятисекундным запросом. Изменение кода касается только выбора SVG.

Единственный запрос на изменение, воспроизведенный этой кампанией, — обновление сеанса — был выполнен, а затем отменен. **В этом отчете измеряется SQL; он не измеряет загрузку или полную загрузку браузера.** Время включает полное получение результатов mysqli, включая примерно 1 МБ SVG.

Запрос от клиента	До (мс)	После (мс)
1. Прочтите сеанс входа в систему	0,34	0,34
2. Обновить сеанс, затем отменить	0,16	0,19
3. Загрузите 61 график и их размеры	5 044,59	3,32
4. Список клиентов	0,53	0,29
5. Список сред	0,78	0,15
6. Список тегов	0,60	0,22

Запрос от клиента	До (мс)	После (мс)
7. Найдите ленту навигации	1,59	0,64
8. Создайте меню	5,26	4,24
9. Подсчет активных тревог	9,21	0,85
10. Выберите пункт меню	0,78	0,66
11. Проверьте ключ перевода	0,53	0,15
12. Читать перевод	0,48	0,18
13. Проверьте таблицу конфигурации	0,20	0,17
14. Чтение настроек лицензии	0,17	0,15
15. Проверьте таблицу скрытых баннеров	0,18	0,15
16. Прочтите маскировку лицензионного баннера	0,30	0,13
Итого, медиана суммы за запуск	**5 068,52**	**12,92**

Анонимизированный отчёт SQL с диапазонами времени и двумя планами дополняет эту таблицу.

Время скачивания у клиента: допущение для сравнения

Для завершения сравнения HTTP мы **берём время загрузки, измеренное на тестовом экземпляре с сервера**, представленное в следующем разделе. Мы предполагаем, что эти продолжительности идентичны для клиента. **Эти значения являются общими для обеих сред; они не представляют собой новую меру HTTP на клиенте.**

Скачивание HTML-документа, повторно используемое как допущение	До	После
Медиана Chromium на тестовом экземпляре	42,90 ms	37,45 ms

Время до первого байта и полной загрузки у клиента повторно не измерялось. Итоговые времена тестового экземпляра использовать нельзя: до исправления один SVG-запрос клиента занимал пять секунд. Измеренный SQL и предполагаемую передачу мы оставляем раздельными и не придумываем время браузера.

Загрузка и браузер: новые измерения с сервера

Чтобы исключить неопределённость из-за соединения, использованного для прежних замеров, **10 октября около 13:45** провели ещё одну серию с сервера Debian 12 до тестового экземпляра в локальной сети. Использовались та же страница «Архитектура» без фильтра, та же временная учётная запись только для чтения, **30 графиков**, окно 1440 × 900 пикселей и Chromium 152. HTTP-кеш браузера отключили; по одному прогревочному запуску исключили, затем сохранили **десять загрузок до и десять после**.

Целевой сервер использует MariaDB 11.8.9 и FrankenPHP / PHP 8.5.9. Код приложения соответствует коммиту `4207dfbe5e7d1709dbbd2efdd2cff8c1e6619dc1`; для варианта «до» только `Dot3LatestGeneration.php` взят из родительского коммита `366b200703ff7550ec01d33e2c7d49c0b570ab15`. Измерения проходят через временный HTTPS-сервер, обслуживающий worktree в локальной сети. Производство остаётся на исправленном коде.

Медианы в миллисекундах:

Этап, измеряемый в Chromium	**До (`EXISTS`)**	**После (`LIMIT 1`)**
Первый байт, PHP и транспорт включены	62,35	50,90
Скачать HTML-документ	42,90	37,45
Документ получен → событие `load`	416,05	392,20
Полная загрузка → `load`	**523,15**	**488,40**



Скачивание документа занимает от **33,2 до 61,3 мс до** и от **32,3 до 55,1 мс после**. Полная загрузка занимает от **472,3 до 681,5 мс до** и от **444,5 до 669,7 мс после**. Диапазоны пересекаются: на этом сетевом пути загрузка длится около половины секунды, без доказанного воспроизводимого выигрыша 34,75 мс в браузере.

Все 30 показанных SVG имеют одинаковый SHA-256 при каждой из двадцати загрузок. Размер распакованного документа — **2 240 909 байт** в обоих вариантах. Сжатое тело занимает около **671 254 байт**, или примерно 671 554 байта с заголовками, которые учитывает Chromium. Этот объём HTML и 30 графиков относятся к тестовому экземпляру и не соответствуют 995 307 байтам SVG для 61 графика у клиента.

Отдельная проверка с помощью `curl --compressed`, десять запросов на вариант после нагрева, подтверждает, что передача остается переменной: медианная загрузка **70,85 мс до и 93,80 мс после**, полный HTTP-ответ **119,97 и 145,62 мс**. `curl` не загружает ресурсы страницы и не выполняет JavaScript. Распаковка, HTTP-ридер, согласованный протокол и нагрузка на сервер способствуют расхождениям с Chromium; эти различия не доказывают регресс SQL.

Контрольные точки измеряются для каждого посещения, а затем суммируются отдельно: **сумма медиан не обязательно является медианой общего числа**. Время после получения документа включает в себя другие ресурсы, анализ HTML, JavaScript и

рендеринг. Мы не связываем это только с расположением графиков. Первый байт уже включает подготовку PHP; расчетное время PHP не добавляется к общей сумме.

Прежние времена загрузки 12 948, 2 952 и 843 мс остаются историческими наблюдениями от 9 октября, доступными в архивном CSV. Скачивание от 470 до 535 мс зависело от нестабильного соединения того дня и больше не служит сетевым ориентиром. Новая серия с 30 графиками не воспроизводит время браузера для экземпляра с 61 графиком.

Новое сравнение SQL на тестовом экземпляре, десять чередующихся пар в одном снимке, доступном только для чтения, с отключенным для сеанса кешем запросов MariaDB, дает **6,94 мс до и 5,81 мс после**. Полные результаты и **2 172 396 байт SVG** идентичны. Первоначальный план там по-прежнему использует `FirstMatch`: даже с 6 591 509 строками в `dot3_cluster` этот экземпляр не воспроизводит дорогостоящую материализацию другого сервера. Поэтому одного только размера таблицы недостаточно, чтобы предсказать медлительность.

Тестовый экземпляр: подробности 16 запросов от 9 октября

Подробная таблица ниже сохраняет серию измерений от **9 октября**: страница без фильтра, учётная запись только для чтения и **30 графиков**. Первый прогревочный запуск исключён, затем для каждого варианта сохранено пять загрузок. Таблица не представляется как новое измерение 16 запросов от 10 октября.

Профилировщик приложения фиксирует **16 запросов при каждой из десяти загрузок**. Каждая строка ниже соответствует запросу, выполненному один раз до и один раз после; последняя строка показывает итог. Указаны медианы времени SQL-вызовов, измеренные на стороне драйвера PHP в миллисекундах. Это не только внутреннее время движка MariaDB.

Запрос	До (мс)	После (мс)
1. Прочтите сеанс входа в систему	0,31	0,34
2. Обновить сеанс	0,29	0,27
3. Загрузите 30 графиков и их размеры	5,86	5,57
4. Список клиентов	0,37	0,34

Запрос	До (мс)	После (мс)
5. Список сред	0,23	0,27
6. Список тегов	0,23	0,29
7. Найдите ленту навигации	0,71	1,07
8. Создайте меню	3,02	3,04
9. Подсчет активных тревог	1,11	1,09
10. Выберите пункт меню	0,67	0,67
11. Проверьте ключ перевода	0,33	0,31
12. Читать перевод	0,30	0,29
13. Проверьте таблицу конфигурации	0,33	0,31
14. Чтение настроек лицензии	0,22	0,23
15. Проверьте таблицу скрытых баннеров	0,32	0,31
16. Прочтите маскировку лицензионного баннера	0,18	0,22
Всего, медиана	**14,53**	**14,59**

Итоговая сумма представляет собой медиану сумм, рассчитанных для каждой нагрузки. Сложение медиан каждой строки не обязательно дает это значение.

В этом тестовом экземпляре **значительного общего прироста SQL не продемонстрировано**. Разница в 0,06 мс минимальна. Это объясняется: MariaDB уже выбрала хороший план для старого запроса. Исправление касается дорогостоящего плана, обнаруженного в производственном экземпляре.

В это число входят запросы приложения при первоначальном рендеринге PHP по трем соединениям, отмеченным профилировщиком. Он исключает последующие вызовы AJAX и внутренние команды входа в систему. Эти 16 запросов не следует назначать всем профилям пользователей или всем установкам.

Также убедитесь, что результаты остаются верными.

Более быстрая страница полезна только в том случае, если на ней отображаются правильные графики. Проверка была сосредоточена на равенстве результатов до и после, готовых или отсутствующих поколениях и запасных решениях.

Девять сценариев были выполнены с использованием тестовых данных, определенных в CTE MariaDB, без вставки в таблицы приложения. Сравнения SQL также охватывали нефильтрованную страницу, фильтрованную страницу и агрегацию графов, при этом результаты сравнивались в одном и том же моментальном снимке транзакций. Тест Placement JavaScript проверил поведение на экранах различной ширины.

Отдельная серия из пятнадцати парных измерений на тестовом экземпляре выявила небольшие медианные накладные расходы на фильтрованном и агрегированном путях: около **0,34 мс** и **0,16 мс**. Мы включаем их в итог: исправление ускоряет не все сценарии. Эти целевые проверки не заменяют полный набор тестов продукта.

Повторить диагностику на другой странице

Начните с разделения трех вопросов: как долго сервер готовит ответ, сколько времени занимает доставка документа и сколько времени требуется браузеру для завершения загрузки?

Затем посчитайте запросы и измерьте каждый. Для того, кто доминирует, сравните план до/после и равенство результатов. В частности, проверьте количество искомых записей, промежуточные результаты и эффективное использование индексов. Повторите измерения: одно наблюдение может ввести в заблуждение.

В этом случае **запрос остался запросом**, а индекс остался прежним. Изменился способ поиска уже имеющихся в кэше рисунков.

Подробные данные и SQL

- 16 запросов, их расположение в коде и полный SQL до/после
- Десять инструментированных нагрузок: продолжительность каждого запроса в CSV
- Двадцать загрузок Chromium от 10 октября с сервера в формате CSV
- Проверка HTTP через curl от 10 октября, до и после
- SQL-сравнение от 10 октября на тестовом экземпляре: результаты и планы в JSON

- Анонимизированный оператор SQL от клиента: 16 запросов, диапазонов и планов
- Анонимизированная конфигурация тестового экземпляра MariaDB `.cnf` и активные переменные, версия и включенные файлы
- Схема 16 таблиц в SVG, определения SQL, перечень двигателей и размеров в CSV и столбцы, индексы и связи в JSON
- сценарий SQL, представленный в тексте, скрипт Chromium, скрипт HTTP и временная конфигурация HTTPS-сервера
- Метод и этапы воспроизведения кампании 10 октября
- Архив: три производственные загрузки от 9 октября на нестабильном соединении

Идентификаторы сеанса и учетной записи заменены параметрами в загружаемых примерах. Производственные и испытательные измерения разделены, что позволяет интерпретировать их, не смешивая две среды.